UNCLASSIFIED

AD NUMBER
ADB028910
LIMITATION CHANGES
TO: **Approved for public release; distribution is unlimited. Document partially illegible.**
FROM: **Distribution authorized to U.S. Gov't. agencies only; Test and Evaluation; JUN 1978. Other requests shall be referred to Air Force Rome Air Development Center, IRAE, Griffiss AFB, NY 13441. Document partially illegible.**
AUTHORITY
RADC, USAF ltr, 26 sep 1980

THIS PAGE IS UNCLASSIFIED

THIS REPORT HAS BEEN DELIMITED

AND CLEARED FOR PUBLIC RELEASE

UNDER DOD DIRECTIVE 5200.20 AND

NO RESTRICTIONS ARE IMPOSED UPON

ITS USE AND DISCLOSURE.

DISTRIBUTION STATEMENT A

APPROVED FOR PUBLIC RELEASE;

DISTRIBUTION UNLIMITED.

LEVEL II

ADB028910

RADC-TR-78-143
Final Technical Report
June 1978

INTERACTIVE PROGRAMMING AND ANALYSIS AIDS (IPAA)

Mr. D. W. Johnson

Harris Corporation ESD

Distribution limited to U.S. Government agencies only; test and evaluation; June 1978. Other requests for this document must be referred to RADC (IRAE), Griffiss AFB NY 13441.

ROME AIR DEVELOPMENT CENTER
Air Force Systems Command
Griffiss Air Force Base, New York 13441

78 07 27 008

DISCLAIMER NOTICE

RADC-TR-78-143 has been reviewed and is approved for publication.

APPROVED: *[signature]*

JOSEPH CAMERA
Project Engineer

APPROVED: *[signature]*

HOWARD DAVIS
Technical Director
Intelligence and Reconnaissance Division

FOR THE COMMANDER: *[signature]*

JOHN P. HUSS
Acting Chief, Plans Office

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: RADC-TR-78-143	2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER	
4. TITLE (and Subtitle): INTERACTIVE PROGRAMMING AND ANALYSIS AIDS (IPAA).	5. TYPE OF REPORT & PERIOD COVERED: Final Technical Report. Sep 77 – Apr 78.
	6. PERFORMING ORG. REPORT NUMBER: N/A
7. AUTHOR(s): Mr. D. W. Johnson	8. CONTRACT OR GRANT NUMBER(s): F30602-77-C-0210
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Harris Corporation ESD, P.O. Box 37, Melbourne FL 32901	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: 62702F, 4027, 0533
11. CONTROLLING OFFICE NAME AND ADDRESS: Rome Air Development Center (IRAE), Griffiss AFB NY 13441	12. REPORT DATE: June 1978
	13. NUMBER OF PAGES: 109
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): Same	15. SECURITY CLASS. (of this report): UNCLASSIFIED
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE: N/A

18 19 6 9 10 15 16 17 05 11 12 108p.

16. DISTRIBUTION STATEMENT (of this Report)

Distribution limited to U.S. Government agencies only; test and evaluation; June 1978. Other requests for this document must be referred to RADC (IRAE), Griffiss AFB NY 13441.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

Same

18. SUPPLEMENTARY NOTES

RADC Project Engineer: Joseph Camera (IRAE)

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Computer Graphics	Modeling
Diagram	Analog Simulation
Interactive Graphics	Interactive Analysis

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A sophisticated, previously-developed concept of an Interactive Analysis Automated Aid has been successfully installed on a Univac 1110 Computer System. This aid is capable of significantly enhancing the analyst's response by provision of a highly interactive means for: (a) modeling new analysis problems as diagrams, (b) executing the analysis diagrams and (c) evaluating the execution results.

Analysis diagrams can be created, edited, modified, stored, (Cont'd)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

78 07 27 008

408972 LB

20 (Cont'd)

retrieved, annotated, and executed at the analyst's graphics station. The diagrams themselves are mathematical in nature to provide the widest flexibility and applicability. Available special features include: handling multipage diagrams, parallel analysis processes, nonlinear analysis and automated input and output of sampled data.

PREFACE

This report presents the results of the Interactive Programming and Analysis Aid Implementation. The report addresses the specific capabilities of an interactive computerized graphics system which merges the strength of an analyst with that of the computer to perform analysis of a broad class of problems.

This report was written and the design and implementation of the software system was done under the direction of Mr. D. W. Johnson, Harris ESD, Melbourne, Florida.

TABLE OF CONTENTS

Paragraph	Title	Page
1.0	INTRODUCTION	1
1.1	Objective	1
1.2	General Nature of Analysis	1
2.0	CONSIDERATIONS IN THE DESIGN OF ITAS	4
2.1	Analyst	4
2.2	Basic Algorithms	4
2.3	Diagram Formation	4
2.4	Diagram Construction Errors	5
2.5	Man/Machine Interaction	5
3.0	DESIGN OF THE IPAA SYSTEM	6
3.1	IPAA and Its FTD Environment	6
3.2	Data Files	8
3.2.1	Display Data File	8
3.2.2	PIF Data File	8
3.2.3	User Data Files	8
3.3	Building Blocks	9
3.4	Graphics Module Functions	9
3.4.1	Diagram Editing	9
3.4.1.1	Add Block	9
3.4.1.2	Block Terminus Points	19
3.4.1.3	Block Connection	19
3.4.1.4	Add Text	19
3.4.1.5	Modify Parameters	19
3.4.1.6	Erase Pin Connection	20
3.4.1.7	Erase Network	20
3.4.1.8	Erase Block	21
3.4.1.9	Erase Text	21
3.4.2	Diagram Storage and Retrieval	21
3.4.2.1	Save Diagram	22

TABLE OF CONTENTS (Continued)

Paragraph	Title	Page
3.4.2.2	Recall Diagram	22
3.4.3	Paging	23
3.4.4	Zooming	23
3.5	Analysis Execution Functions	25
3.5.1	Process Control Parameter Modification	25
3.5.2	IPAA Data Base	28
3.5.3	Block Ordering	28
3.5.4	Interpolation	31
4.0	COMPARATIVE ALGORITHMS	32
4.1	Algorithm Number One	32
4.1.1	Algorithm No. 1 - Inputs and Mathematical Formulation	33
4.1.2	Algorithm No. 1 - Comparison	35
4.2	Algorithm Number Two	36
4.2.1	Algorithm No. 2 - Mathematical Formulation	36
4.2.2	Algorithm No. 2 - Comparison	40
5.0	CONCLUSIONS	43
6.0	RECOMMENDATIONS	44
6.1	Macro Capability	45
6.1.1	Macro Definition	45
6.1.2	Macro Implementation	45
6.2	Graphic Display of Output Data	49
6.3	Current System Modifications	49
6.3.1	Uniscope Execution	49
6.3.2	Addition of Functional Blocks	49
7.0	REFERENCES	51
APPENDIX A	UTILIZATION OF LINES AS BLOCK CONNECTORS	A-1
APPENDIX B	ALGORITHMS LISTINGS AND DIAGRAMS	B-1

EVALUATION

This report describes a unique approach to solving the problem of inadequate man-machine interaction achieved by the conventional approach to coding scientific computer algorithms. The report examines the utilization of a graphical block diagram schematic approach to programming which enables a user to solve complex engineering problems interactively with the aid of a computer. This approach was evaluated through specific examples and proved to be a very powerful tool in the hands of trained analysts.

In addition, this report describes recommended modifications/additions to the IPAA Software System for future investigation.

JOSEPH CAMERA
Project Engineer

1.0 INTRODUCTION

This report describes the implementation of the Itas concept on the Univac 1110 Computing System at FTD, Wright Patterson AFB. This particular implementation of the Itas concept will be referred to as IPAA in the report. Background information on the Itas concept is also included for completeness.

1.1 Objective

The contract objective is to implement the Itas concept on the Univac 1110 Computing System which has Tektronix 4014 graphics terminals. This general-purpose computing system supports a wide range of analysis activities, and as such, the analysts have no standard data or data format. Therefore, a secondary contract objective is to allow the analyst to access a variety of data formats (both input and output) within the IPAA software system.

1.2 General Nature of Analysis

Most analysis activities can be based upon description of the problem under investigation through the use of a schematic diagram. Examples of such diagrams are plentiful; logic networks, flow charts, electronic schematics, mechanical diagrams, system diagrams, etc. Each of these schematic diagrams are basically topological graphs which are drawn to be readable and suggestive to the analyst trained in the particular discipline. As a result, the diagrams serve as primary vehicles for analyses, communication, instruction and reporting of results. Not only do these diagrams conveniently describe the problems to a trained analyst, but they are interpretable by the computer as they are drawn, thereby providing a high level man/machine interface "language".

For the broad class of analysis problems considered above, the general methodology of analysis can be given in terms of the six phases of activity shown in Figure 1 . Although only the primary flow is shown, other flows are also possible as it is common practice for an analyst to

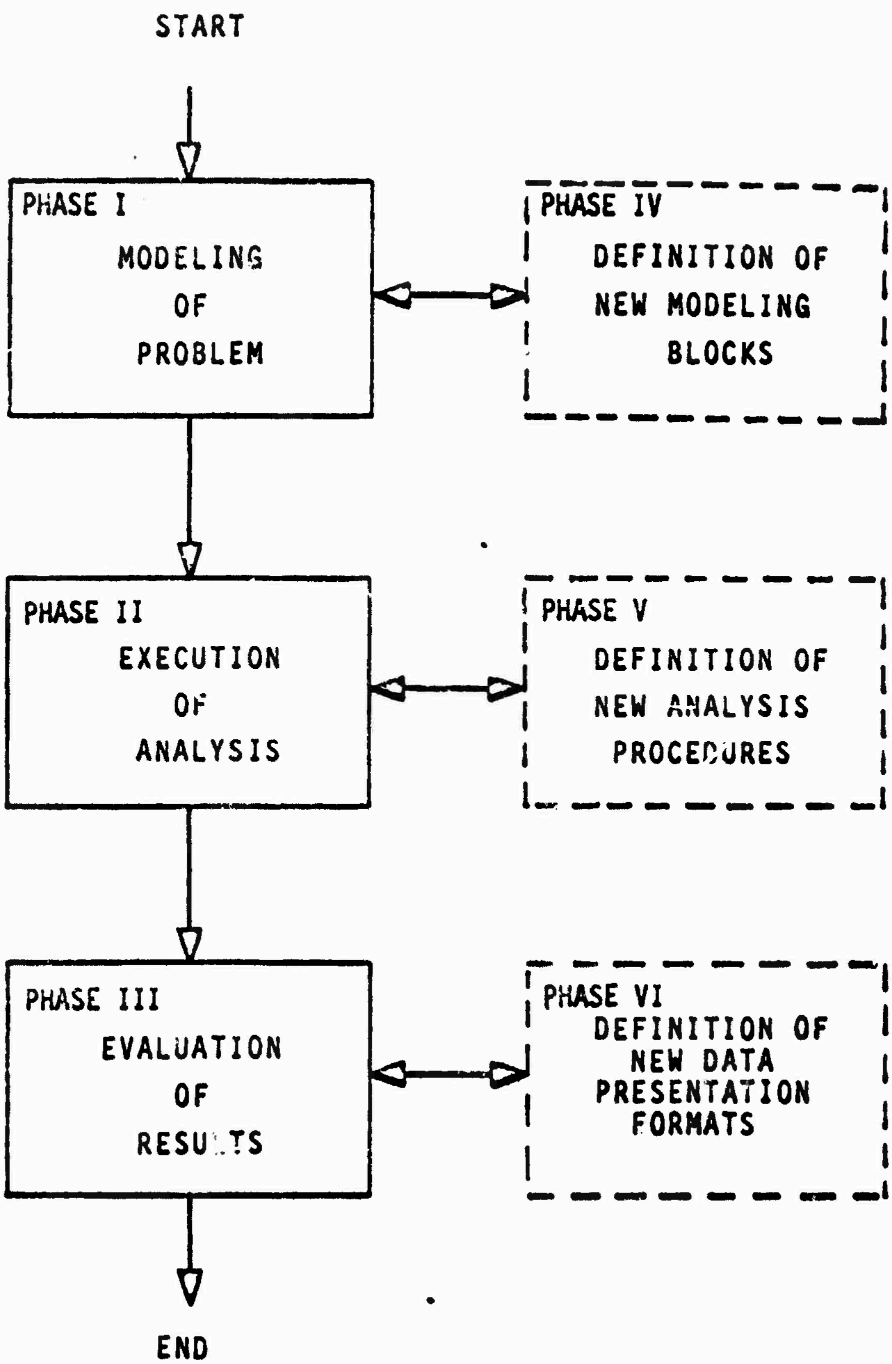


Figure 1.

The Six Phases of Analysis Methodology

backtrack to an earlier phase of activity to change the nature of the problem analysis in some fashion.

The general nature of analysis methodology is normally based on three consistently used phases:

I. Modeling of Problem

II. Execution of Analysis

III. Evaluation of Result

Less consistently utilized are the three support phases:

IV. Definition of new modeling block

V. Definition of new analysis procedures

VI. Definition of new data presentation formats

In order to understand the flow shown in Figure 1, assume that the analysis is electrical in nature. The analyst will then think of modeling his problem in terms of electrical circuits (Phase I). He will normally use the basic modeling blocks (in modeling theory called connectives, primitives, symbols or tools) of resistors, capacitors, inductors, current sources and voltage sources. Occasionally, he might be forced to create new modeling blocks (Phase IV) such as a nonlinear element. Once the problem has been modeled as a schematic circuit diagram, the analyst will analyze it (Phase II). This analysis will likely involve calculating the response to a particular input signal. In the process, the analyst may be required to develop new analysis procedures (Phase V) such as sensitivity analyses, transient analyses, controllability analyses, etc. Upon completion of analysis calculations, the analyst will evaluate the results of the analysis (Phase III). To do this, he will normally display the results in some manner such as waveform plots. On other occasions, he may devise other data presentation formats (Phase VI) such as Bode plots, histograms, etc., prior to evaluating the results.

2.0 CONSIDERATIONS IN THE DESIGN OF ITAS

Prior to the design of Itas certain desirable features were innumerated and carefully considered. Also taken into account in the Itas design was the important fact that Itas-aided analysis would be performed by specialized users in a restricted environment.

2.1 Analyst

In general, an analyst will not be a trained computer programmer though, most probably, he will have had significant exposure to computer programs. Thus the procedures of diagram generation and analysis execution should not be dependent on knowledge of computer techniques, computer language or computer jargon.

2.2 Basic Algorithms

The set of basic algorithms or blocks should encompass all the fundamental functions that an analyst might reasonably require in the constructions of analytic processes relating to the analysis of telemetry data. On the other hand, the set of primitive blocks should be small enough to be displayed on a small fixed portion of the CRT screen. Also the identification of the primitives on the CRT screen should be readily recognizable and there should be no ambiguity in the meaning or use of the blocks.

2.3 Diagram Formation

The interconnecting of the primitive blocks to form a processor should be a fairly simply and logical procedure. It is also reasonable to expect the analyst will wish to make changes in the diagram after he has examined the results of the analysis. Therefore, relatively minor changes should not require a complete redoing of the diagram. Should the user wish to make diagram changes while he is in the process of generating the diagram, the changes should be simple to incorporate.

2.4 Diagram Construction Errors

Inconsistencies in the construction of a diagram should be flagged and rejected by the program. The user should be able to continue after an error without undue interruption and with no loss in that portion of the diagram already completed.

2.5 Man/Machine Interaction

A most significant feature of Itas should be the merger of man's peculiar capabilities with those of a digital computer. For instance, a man can think in broad terms, can reason intuitively, can remember previous trials and benefit from them. On the other hand, the computer has great speed and accuracy and can do complex mathematical computations.

3.0 DESIGN OF THE IPAA SYSTEM

The IPAA Software System is designed to functionally adhere to the Itas concept; i.e., to provide basic tools to the analyst which he may use to create unique analysis sequences to process real or simulated data, existing in a variety of formats.

IPAA is made up of two functional subsystems; a set of graphics routines which the analyst uses to compose processing algorithms and a set of analysis execution routines which process the data in accordance with the previously generated algorithms.

3.1 IPAA and Its FTD Environment

IPAA software exists as a stand-alone program within the Univac 1110 executive operating system. IPAA is accessed by performing the instructions displayed in Figure 2.

More specifically, the IPAA System possesses the following individual features which are designed to improve the Itas concept in the Univac 1110 FTD working environment:

1) IPAA is designed to operate on the Univac 1110 host computer and operate within the constraints of the executive time-sharing oriented operating system.

2) The IPAA System makes full use of the Tektronix 4014 graphic display terminal as the prime means of analyst interaction. The IPAA software interfaces with the Tektronix 4014 terminal through the use of the TCS graphics display software package.

3) The IPAA software is coded in the Fortran V compiler language. The use of Fortran has two important benefits:

STEP 1 - SIGN ON TEKTRONIX TERMINAL WITH NORMAL
FTD SIGN ON PROCEDURES.

STEP 2 - TTY LOCK

STEP 3 - @XQT FTD$*IPAA.IPAA

Figure 2. ACCESSING THE IPAA SOFTWARE

a) The software will be easy to maintain and/or modify.

b) The programs are self documented by the generous use of comment cards inserted in a uniform and precise manner.

4) The IPAA software is generally the result of a structured top/down system design. The resulting software is modularized according to specific functions and/or services. The software developed is thus easier to maintain.

3.2 Data Files

There are three types of data files closely associated with the IPAA System. In general, storage capability is provided for the diagram, processing instructions, and the input/output data.

3.2.1 Display Data File

All information associated with a diagram will be stored within a user-named data file. This includes all blocks, function codes, connections, and parameters of the nine full pages of display in a single data file.

3.2.2 PIF Data File

All user instructions pertaining to the format of the input and output data files and of the execution control parameters will be stored in the processor information file (PIF).

3.2.3 User Data Files

The specified user data files may or may not have been generated by the IPAA software system. The IPAA software system, however, does provide for the input and output of data in a great variety of formats.

3.3 Building Blocks

The IPAA diagram is based on interconnecting a set of primitive blocks most of which are mathematical operators. This mathematical structure being universal has direct and immediate utility for a variety of analysis disciplines. An IPAA graphics menu as shown in Figure 3 has a list of all of the primitive blocks in the upper right portion of the display page. Figure 4 is a description of each block including its inputs, outputs and parameters.

3.4 Graphics Module Functions

The graphics module provides the capability for an analyst to build data-processing diagrams on an interactive basis with the computer. The module maintains all diagram-associated data in a sophisticated database which is transferred to the analysis execution module when the diagram is to be executed. The module also saves diagrams for subsequent execution and/or modification.

3.4.1 Diagram Editing

The generation of a complete diagram is the process of combining the various diagram elements (blocks, lines, parameters, and text) into an executable entity. The editing function as considered here includes not only the addition of elements to a partially constructed diagram but also the deletion of unwanted elements that are a part of the diagram. The various editing functions are discussed in the following paragraphs under Section 3.4.1.

3.4.1.1 Add Block

The addition of a block requires, in general, two distinct operator actions; selection of the add block function and selection of the block's operational function. Selection of the add block function in IPAA automatically specifies screen position. While not necessarily aesthetically appealing, each block was chosen to be the same in size and shape and to be distinguishable only by an up to 6-character mneumonic

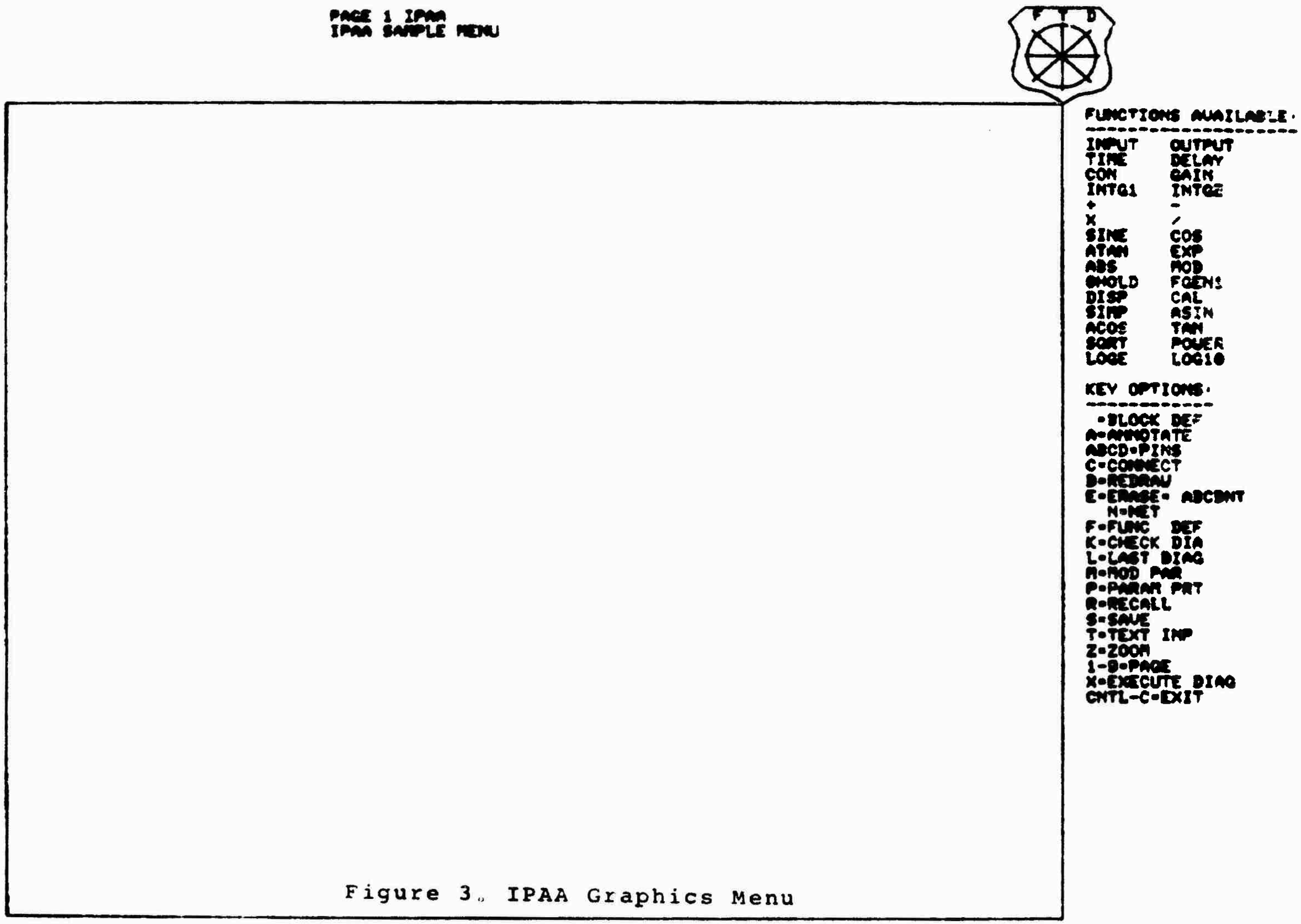


Figure 3. IPAA Graphics Menu

BLOCK TYPE	BLOCK FUNCTION	NO. OF PARAMETERS	NOTES
INPUT	1. Enter data from a data file or 2. Used as a diagram page connector.	1	1. When used to bring data from a file, the parameter value must lie between +1 and +4 for default format, and between +1 and +6 for all other formats. 2. When used as a page connector, the parameter value must be negative and equal to the output block to which it is connected.
OUTPUT	1. Writes data to a designated output file. 2. Used as a diagram page connector.	1	1. When used to output a file of data, the parameter value must lie between +1 and +4 default format, and between +1 and +6 for all other formats. 2. When used as a page connector, the parameter value must be negative.
TIME (t)	Output the current time	0	The start time, stop time and time increment is specified during the Analysis Execution phase.

Figure 4. FUNCTIONAL BLOCK DEFINITIONS

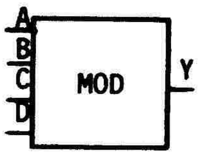
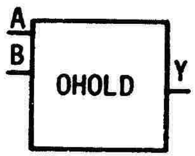
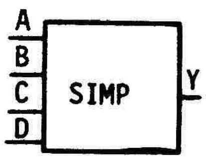
BLOCK TYPE	BLOCK FUNCTION	NO. OF PARAMETERS	NOTES
	Computes the real valued modulus	1	Parameter 1 = modulus Y = AMOD [(A*B) + (C*D), P1]
	Hold value until directed to change	1	Parameter 1 = fixed hold value If A 0 Y=0 A=0 Y=P1 A 0 Y=B
	Perform Simpson's Rule Integration	3	$X_n = A_n+B_n+C_n+D_n$ P_1 = INITIAL VALUE = Y_o or Y_{n-1} $P_2 = X_{n-2}$ $P_3 = X_{n-1}$ $Y_n = Y_{n-1}+\Delta t*\left[X_{n-2}+4*X_{n-1}+X_n\right]/6$

Figure 4 (Continued)

BLOCK TYPE	BLOCK FUNCTION	NO. OF PARAMETERS	NOTES
DELAY (input X, output Y)	Provide a one cycle delay	1	P = Initial value Y = P P = X
CON (output Y)	Output a constant value	1	Y = C where C is entered as the parameter
GAIN (inputs A, B, C, D, output Y)	Multiplies the sum of a series of inputs by a constant	1	Y = G * (A+B+C+D) where the parameter is the gain G
INTG1 (inputs A_n, B_n, C_n, D_n, output Y_n)	Trapezoidal integrator	2	Parameter 1 = initial value out of integrator (Y_{n-1}) Parameter 2 is X_{n-1} where $X_n = A_n + B_n + C_n + D_n$ Y_o = Parameter 1 X_o = Parameter 2 $Y_n = Y_{n-1} + \Delta t/2(X_n + X_{n-1})$ where Δt = time between successive points

Figure 4 (Continued).

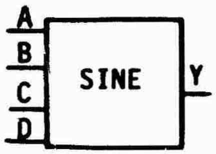
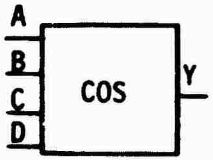
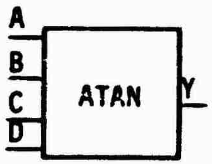
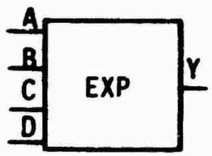
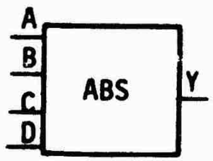
BLOCK TYPE	BLOCK FUNCTION	NO. OF PARAMETERS	NOTES
	Computes the sin	0	$Y = \sin\left[(A*B) + (C*D)\right]$
	Computes the cosine	0	$Y = \cos\left[(A*B) + (C*D)\right]$
	Computes the principal value of the arctangent	0	$Y = \tan^{-1}\left[(A*B) + (C*D)\right]$
	Computes the exponential function	0	$Y = \mathrm{EXP}\left[(A*B) + (C*D)\right]$
	Computes the absolute value	0	$Y = \left\|\left[(A*B) + (C*D)\right]\right\|$

Figure 4 (Continued).

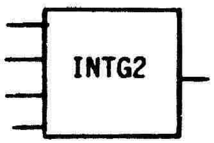
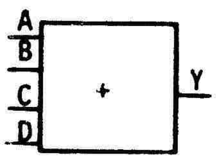
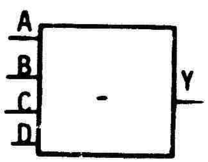
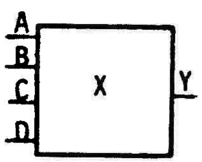
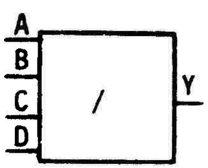
BLOCK TYPE	BLOCK FUNCTION	NO. OF PARAMETERS	NOTES
	Not implemented		
	Adder	0	Y = A + B + C + D
	Subtractor	0	Y = A - B - C - D
	Multiplier	0	Y = A * B * C * D
	Divider	0	Y = A/B/C/D

Figure 4 (Continued).

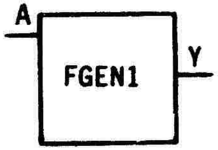
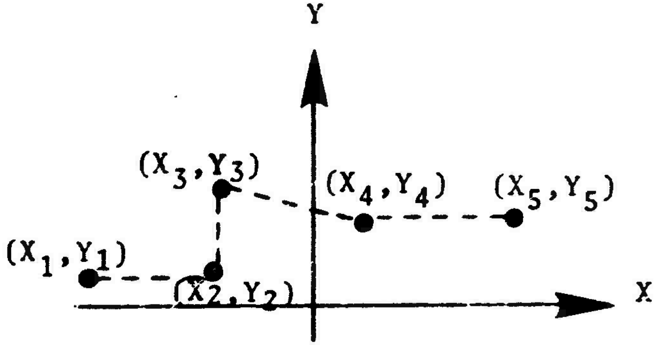
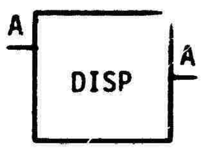
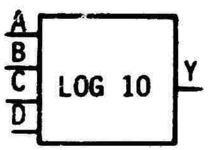
BLOCK TYPE	BLOCK FUNCTION	NO. OF PARAMETERS	NOTES
	Piecewise linear interpolation	11	P1 = number of pairs. Parameters 2-11 are used in pairs to define the interpolator boundaries
			Y = f(A, P2---P11) where X_1 = P2 Y_1 = P3 X_2 = P4 Y_2 = P5 X_3 = P6 Y_3 = P7 X_4 = P8 Y_4 = P9 X_5 = P10 Y_5 = P11
	Display the output value, A, of a block	0	
	Computes the common log	0	$Y = \text{Log}_{10}\left[(A*B) + (C*D)\right]$

Figure 4 (Continued).

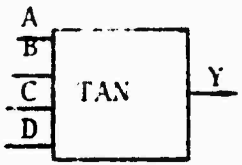
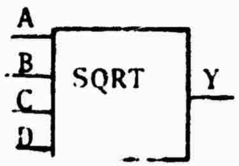
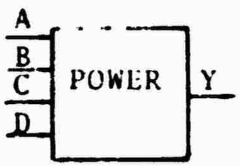
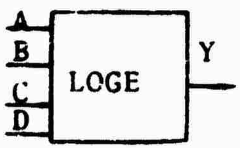
BLOCK TYPE	BLOCK FUNCTION	NO. OF PARAMETERS	NOTES
	Computes the tangent function	0	$Y = \tan\ [(A*B) + (C*D)]$
	Computes the square root	0	$Y = \sqrt{(A*B) + (C*D)}$
	Computes a real number raised to a power	1	$Y = [(A*B) + (C*D)]^{P1}$
	Computes the natural log	0	$Y = \mathrm{Log}_e\ [(A*B) + (C*D)]$

Figure 4 (Continued).

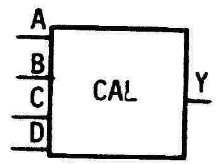
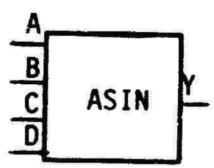
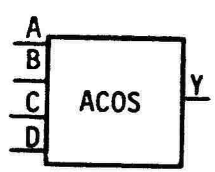
BLOCK TYPE	BLOCK FUNCTION	NO. OF PARAMETERS	NOTES
	Performs elementary arithmetic functions based on parameter values	3	Parameter 1 defines f (A,B) Parameter 2 defines G[f(A,B),C] Parameter 3 defines H{G[f(A,B),C],D} When the Parameter value is: 1 the operation is addition 2 the operation is subtraction 3 the operation is multiplication 4 the operation is division.
	Computes the arcsin function	0	Y = arcsin [(A*B) + (C*D)]
	Computes the arccosine	0	Y = arccos [(A*B) + (C*D)]

Figure 4 (Continued).

enclosed in the block. The number of input stubs and parameters are determined by the program from the block's function.

3.4.1.2 Block Terminus Points

Line terminus points on the blocks are critical forms of data items in a diagram and are used in subsequent machine calculations. Terminus points or input/output line stubs are generated on the block schematic when it is displayed. The number of input termini is determined by the block's function. A terminus becomes active in a usable part of the diagram only if a line is connected to it.

3.4.1.3 Block Connection

Functionally, the analyst need only specify the output of a particular block be connected to one of the input pins of the connecting block. At this point, the IPAA software system will generate the connecting line between the two blocks. A detailed discussion of the method of graphically drawing this connection is contained in Appendix A.

3.4.1.4 Add Text

Text may be appended to a block during the construction of a diagram at which time it becomes an integral part of the graphics database. The text serves two purposes; 1) automatically it becomes a significant part of the analysis documentation, and 2) upon subsequent recall of the diagram, it may aid the user in understanding the purpose and function of the diagram. A detailed description of the procedure for adding text is given in Section 3.3.7 in Reference 3.

3.4.1.5 Modify Parameters

Figure 4 shows which of the blocks require parameters and their meaning. Those blocks which require parameters have them all initialized to zero. The procedure for modifying parameters is given in Section 3.3.9 in Reference 3. Although parameters could be permanently displayed

in a diagram, it was felt that a picture would become too busy and so lose its effectiveness. The user may quickly and easily check the current value of the parameter by using the methods described in Section 3.3.8 in Reference 3.

3.4.1.6 Erase Pin Connection

Any input pin connection may be deleted from a finished diagram or from a partially completed diagram. After the user selects the input pin which he wishes to be deleted, the program proceeds as follows:

1) The designed input pin pointer is checked to determine if it has been used. If not, an error is printed for the analyst.

2) If the pin has been used, the software crosses out the input connection on the screen.

3) The input pin pointer is then flagged as having been deleted but still displayed. (Note: should the analyst decide at a later time to connect another block to this particular input pin, the software will allow the connection; however, it will immediately erase the screen and redisplay that page.)

3.4.1.7 Erase Network

Any connection network (where a network consists of all the line connections that eventually connect to a common block output stub) may be deleted from a completed or partially completed diagram. After the user has selected the block from which the output is to be deleted, the program proceeds as follows:

1) The graphics data base is searched to determine which input pin connection pointers are referencing the designated output block.

2) As each pin is determined to have that particular reference, the software then proceeds to perform the functions referenced in Items (2) and (3) in Section 3.4.1.6.

3.4.1.8 Erase Block

Any block may be deleted from a completed or partially completed diagram. The user selects the block to be deleted according to the method described in Section 3.3.10 of Reference 3. The program then proceeds as follows:

1) The indicated block is crossed out by the software on the display.

2) All input pointers and the parameter pointer is zeroed.

3) The software then proceeds to perform the functions referenced in Items (1) and (2) in Section 3.4.1.7.

3.4.1.9 Erase Text

Text may be erased from a diagram during construction, upon completion or after the diagram has been recalled and the editing is desired. A detailed description of the procedure necessary to accomplish this is contained in Section 3.3.13 in Reference 3. The program proceeds as follows:

1) The block location is determined from the screen coordinates sent to the software by the cursor position of the terminal.

2) The indicated text array at this point is then set to blank characters.

3.4.2 Diagram Storage and Retrieval

In order to accomplish the storage or retrieval of a diagram, a transfer of information to or from a disc file is necessary. This

information includes the graphics database and any indicated parameters or text associated with that database.

The IPAA System has a paging capability in which 1-9 pages may comprise a single diagram. The total of nine pages is stored as one disc file.

In keeping with the existing protocol for operating within the Univac 1110 environment, the analyst may assign a 12-character alphanumeric name to a file.

3.4.2.1 Save Diagram

The purpose of the Save Diagram function is to give the analyst the capability of storing diagrams for future use. This storage is accomplished nine pages at a time. The detailed procedure necessary to save a diagram in given in Reference 3, Section 3.3.14. Program interaction is briefly described below:

1) Upon command, the software will request the name of the desired file from the analyst.

2) The file is then created within the Univac 1110 system.

3) The necessary graphics database is then written to that file.

4) The file is then catalogued and is available for later recall by the analyst.

3.4.2.2 Recall Diagram

This function allows the analyst to recall to the screen any diagram he has previously stored. Once a diagram has been recalled, the analyst can either modify the diagram or run the analysis execution function. The procedure necessary to recall a diagram is given in Reference 3, Section 3.3.15. Program interaction is briefly described below:

1) Upon selection of the recall diagram function, the analyst must specify the file name of the diagram to be recalled.

2) The correct file is then located and the information read into the program memory.

3.4.3 Paging

A diagram of the IPAA System may be made up of as many as nine separate display pages. The pages are connected by referring a specially designated output block of one page to a similarly designated input block of another page. This special usage of output and input blocks may be seen in Figure 5. For every diagram created and stored, there will be one data file generated.

The present configuration of IPAA provides for no more than 180 blocks in a complete diagram. When a diagram is continued from one page to another it is necessary to have an output block and an input block as connectors. These blocks are not formally a part of the processing function of the diagram but they are counted as part of the 180 allowable blocks. The inclusion of these blocks was made necessary by the logic of the analysis ordering algorithm as described in Section 3.5.3. Further user-related details of paging are discussed in Section 3.3.1 of Reference 3.

3.4.4 Zooming

In connunction with the paging capability a zooming feature has been incorporated into IPAA. Zooming allows the user to display all nine pages of the diagram at once. Because of the size of the blocks in trying to display 180 blocks on a single display, it will be noted that only the block number is displayed. There is no function code and no parameters. There is, however, an additional paging number (absolute value of the parameter for input and output blocks) which is printed to the left or right of input and output blocks in order to easily reference the interpage connections.

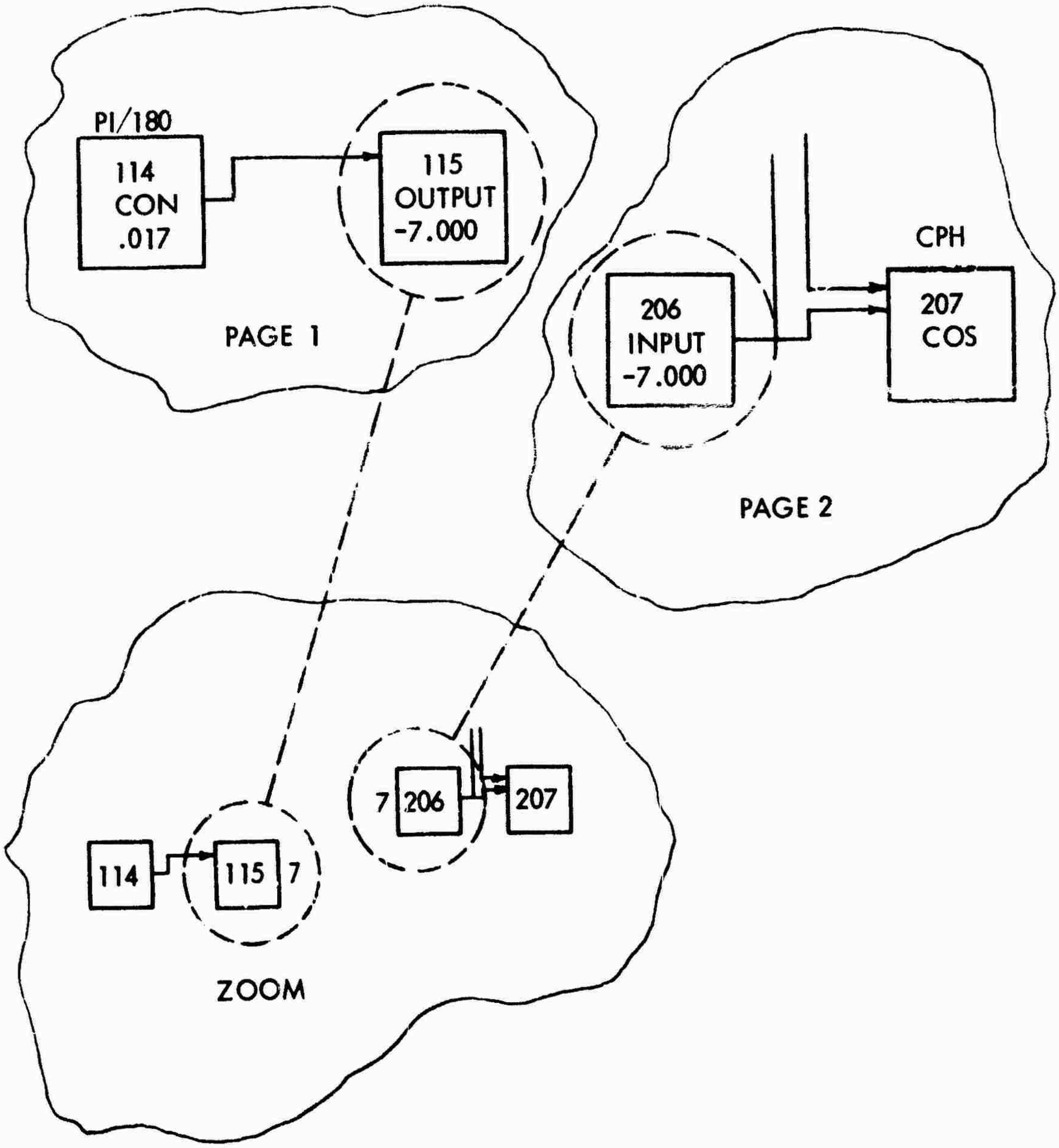


Figure 5. Page Connections

3.5 Analysis Execution Functions

An IPAA diagram as constructed by the graphics function will not be in an executable state. Execution of the diagram requires that the order of operating the various functions of a diagram be established. For example, it would be disastrous to run a subroutine before it's current inputs were available. There are basically two methods of controlling diagram execution, (1) a syntax compiler and (2) a table-driven compiler.

A syntax compiler (not chosen for IPAA) produces a unique coding sequence in some language syntax, such as FORTRAN. The main program would consist of a series of calls to subroutines that are a part of the main program. The program would be required to establish the order of the calls and assure that proper parameters were available to each called subroutine. These calling statements are built in source format and input to a compiler program for final compilation, linking and loading. The output of this compiler is a unique problem-oriented program. A change to the analysis diagram requires an iteration of compiling, linking and loading. This type of system is not suited to diagrams of the complexity processed by IPAA.

The table-driven compiler as used in IPAA consists of a main program and all the subroutines and functions that might possibly be used, not just those required for the particular diagram to be executed. The preprocessor portion of the analysis execution function generates a table that determines the order of execution of the various mathematical subroutines. The run-time portion of the analysis functions scans the run table and calls the appropriate mathematical functions in the proper order.

3.5.1 Process Control Parameter Modification

To execute the analysis of a diagram, the following process control parameters are required:

Start Time
Stop Time
Time Increment
Diagram
Input Interpolation

Display Ratio
Input Descriptions
Output Descriptions

The default values for these parameters are given in Figure 6 which is the analysis execution menu. The analyst may modify any of these parameters according to the procedures given in Reference 3, Section 3.4. The purpose of these parameters is discussed below:

Diagram: The name of the diagram may be provided by the analyst or it may be the default value as seen in Figure 6.

Start Time: This time refers to the start time in a time block of a diagram. Default time is 0.0 which may be changed to any time desired by the analyst.

Stop Time: This value refers to the time at which the analyst wishes to end the analysis run. Default stop time is 10.0 and may be changed to any desired time.

Time Increment: This value is the delta time used in the analysis and may be changed to any delta time increment desired.

Input Interpolation Degree: A Lagrange interpolation function is provided in which polynomials up to a ninth degree may be evaluated. The default value is 1 which would provide a linear interpolation. If the analyst desires a polynomial interpolator, then the degree of the polynomial is entered.

Display Ratio: The display blocks entered in a diagram provides the analyst with the means to view immediate results on the CRT screen. If the analyst does not desire to see these immediate results at every delta time period in the analysis, the display ratio may be modified to any desired time increment the analyst wishes to view.

** IPAA PROCESSOR INFORMATION **

1- PROCESSING DESCRIPTION:		
2- DIAGRAM NAME		
3- START TIME	.000	
4- STOP TIME	10.000	
5- TIME INCREMENT	.100	
6- INPUT INTERPOLATION DEGREE	1	
7- DISPLAY RATIO	1	
8- TIME OPTION	TIME INPUT PROVIDED	
<u>INPUTS:</u>		
9- INPUT FORMAT	DEFAULT FORMAT (POT)	
	FILE NAME	START TIME
10- INPUT FILE NO. 1		.000
11- INPUT FILE NO. 2		.000
12- INPUT FILE NO. 3		.000
13- INPUT FILE NO. 4		.000
<u>OUTPUT</u>		
14- OUTPUT FORMAT	DEFAULT FORMAT (POT)	
	FILE NAME	
15- OUTPUT FILE NO. 1		
16- OUTPUT FILE NO. 2		
17- OUTPUT FILE NO. 3		
18- OUTPUT FILE NO. 4		
20- SAVE PIF IN FILE		
21- RESTORE PIF FROM FILE		
30- EXECUTE		

Figure 6. Default Execution Menu.

Inputs: IPAA provides for a variety of input formats. In addition to the POT-type of input file, the system will accept binary, formatted and free-field inputs. These three options allow mixtures of floating point and integer variables. See Section 3.4.9 and Appendix C in Reference 3 for a complete discussion.

Outputs: IPAA provides essentially the same format variety on output as on input with the exception that the free-field option is not allowed. See Section 3.4.9 and Appendix C in Reference 3 for a complete discussion.

3.5.2 IPAA Data Base

A complete functional description of an IPAA processing block is contained in six words (See Figure 7). This includes function code, input pointers, and parameter list pointer. Thus the complete diagram (180 blocks) may be functionally described in an array of length 6x180. This array is called the block table.

The IPAA data base consists of the block table and a parameter array. The length of the parameter array is fixed at 300, which is deemed to be sufficient for the current software system.

3.5.3 Block Ordering

The ordering algorithm is the heart of the analysis execution function. In addition to the block table and parameter array, the ordering algorithm makes use of a software table in the IPAA system. This table consists of a flag for each available function denoting whether or not it has an output available initially. The current implementation flags only delay and integrator blocks as having outputs initially available. The procedure will now be discussed in a step-by-step manner.

Step 1. Cycle through the block table and remove the off-page connections. This is performed by redirecting the pointers to an input page connector to the block which is input to the output page connectors.

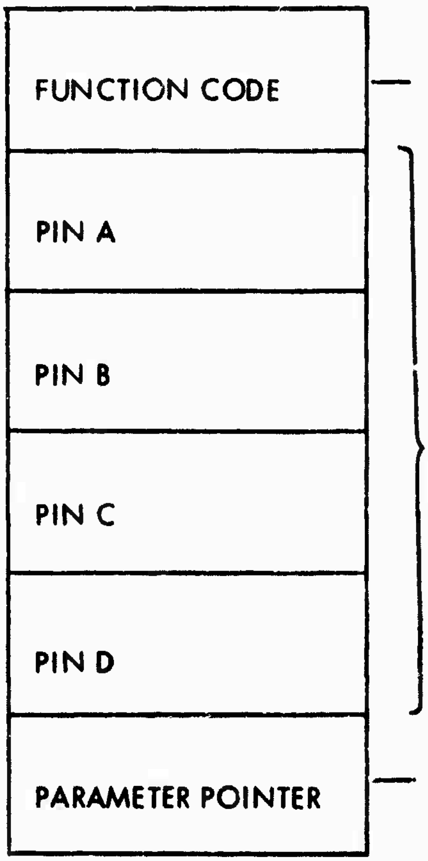
In memory, each block is represented by a 6-word table entry:

KFUN(I,J,K) where I = 1 to 6 (the table entries)

J = 1 to 20 (the block number on the page)

K = 1 to 9 (the page number)

Special codes in these table entries are shown below.



A positive integer M where M varies from 1 to N, and is the index in the table of defined function codes (MATCH). 0 if function is not defined. A negative integer indicates that the block has been used but the function has not yet been defined.

An integer entry containing the block number (1 to 180) from which the input to this pin comes.

For all blocks:
Contains an integer pointing to the first entry (for this block) in the parameter table.
For Input/Output block parameters:
A positive integer indicates the number of the file (cross-referenced in the Processor Information File - PIF) to be used in I/O operations.
A negative integer indicates an off-page connection. I/O blocks with this same negative number are logically connected during execution.

Figure 7. BLOCK FUNCTIONAL DESCRIPTOR

Step 2. Cycle through the block table and put into the ordering table pointers to those blocks that have outputs available and which may be ordered immediately. This set consists of all constant, input, and time blocks.

Then set, in tables parallel to the ordering table, a flag indicating that the block is in the ordering table and another flag indicating the block has its output available.

Step 3. Cycle through the block table and set the appropriate output available flag for each function. This information is derived from the software table.

Step 4. Set the block table pointer to the first entry in the table.

Step 5. Has this entry been ordered? If so, go to Step 8. If not go to Step 6.

Step 6. Do all the blocks referenced by this block's input pointers have outputs available? If yes, go to Step 7. If not go to Step 8.

Step 7. Order the block and flag its output available.

Step 8. Increment the block table pointer. If the table has been exhausted go to Step 9. If not, go to Step 5.

Step 9. Are there any blocks which haven't been ordered? If not, the ordering process is finished. If there are blocks remaining to be ordered and at least one was ordered on the last pass through the table, go to Step 4. If none were ordered on the last pass and there are still blocks remaining to be ordered, then terminate process with error flag.

3.5.4 Interpolation

Provision was made within the Itas concept to handle non-uniformly sampled telemetry formats. A Lagrangian interpolating polynomial algorithm with degree n is provided to handle such non-uniform data. The degree of interpolation (linear, quadratic, etc.) is under analyst control.

The general form of Lagrange's interpolating polynomial is given by

$$P_n(x) = \sum_{i=0}^{n} L_i(x)f(x_i)$$

where

$$L_i(x) = \prod_{\substack{j=o \\ j\neq i}}^{n} \frac{(x-x_j)}{(x_i-x_j)} \quad , \ i=0,1,\ldots n$$

Each functional value, $f(x_i)$, included in the polynomial fit is multiplied by L_i, an n^{th} degree polynomial in x (since there are n factors $(x-x_j)$. For a more detailed mathematical discussion of the Lagrange interpolation polynomial, the reader is referred to References 1 and 2.

4.0 COMPARATIVE ALGORITHMS

Two algorithms were chosen by the customer for comparison of conventional programming techniques versus an IPAA approach.

Although it is felt that for certain types of processing the IPAA approach may be faster and somewhat less error-prone than the conventional FORTRAN program, it is difficult at best to attempt a comparison of coding/debug time (conventional) versus diagram building (IPAA). Such a comparison would apply only to the individuals involved and any conclusions would not necessarily apply to other users. For this reason, the comparisons in this section will be concerned with capability of the different approaches and the computer system knowledge required for implementation.

Common to both algorithm comparisons is the prerequisite that the conventional approach requires a knowledge of the FORTRAN programming language and a programmer's familiarity with the Univac Operating System and file-handling abilities as opposed to the superficial computer-system knowledge required by the IPAA-approach. This means that some analysts may not be able to make use of the computer system except through IPAA due to a lack of training and knowledge in the software disciplines.

4.1 Algorithm Number One

This algorithm takes inertially fixed orthogonal sensed velocities through an Euler angle transformation (as defined by the Gimbal angles and order) to obtain orthogonal sensed velocity in the vehicle body coordinate frame. The RSS of the orthogonal vehicle velocities are computed to obtain total velocity and the orientation of this velocity with respect to the vehicle body is calculated. In addition, differences of the input sensed velocities are used to calculate the orthogonal sensed acceleration in the vehicle body coordinate frame, and the total acceleration and orientation with respect to the vehicle body. (See program listing in Appendix C.)

4.1.1 <u>Algorithm No. 1 - Inputs and Mathematical Formulation</u>

The input variables defined for algorithm one are as follows:

T_i = Time
IG_i = Inner Gimbal Angle
MG_i = Middle Gimbal Angle
OG_i = Outer Gimbal Angle
VV_i = Vertical Sensed Velocity
HV_i = Horizontal Sensed Velocity
LV_i = Lateral Sensed Velocity

Figure 8 illustrates the test case used in the comparison.

Let $\dot{X}$, $\dot{Y}$, $\dot{Z}$ = X, Y, Z velocities

$\dot{MAG}$ = Total Velocity Magnitude

$\dot{\phi}$ = Angle of total velocity relative to vertical longitudinal plane.

$\dot{\gamma}$ = Angle of total velocity relative to horizontal plane.

Then

$$\dot{X}_i = \left[HV_i \cdot \mathrm{Cos}(IG_i) - LV_i \cdot \mathrm{Sin}(IG_i)\right] \cdot \mathrm{Cos}(OG_i) + \left[HV_i \cdot \mathrm{Sin}(IG_i) + LV_i \cdot \mathrm{Cos}(IG_i)\right] \cdot \mathrm{Sin}(OG_i)$$

$$\dot{Y}_i = \left[HV_i \cdot \mathrm{Sin}(IG_i) + LV_i \cdot \mathrm{Cos}(IG_i)\right] \cdot \mathrm{Cos}(MG_i) - VV_i \cdot \mathrm{Sin}(MG_i)$$

$$\dot{Z}_i = \left\{\left[HV_i \cdot \mathrm{Sin}(IG_i) + LV_i \cdot \mathrm{Cos}(IG_i)\right] \cdot \mathrm{Sin}(MG_i) + VV_i \cdot \mathrm{Cos}(MG_i)\right\} \cdot \mathrm{Cos}(OG_i)$$

$$\dot{MAG}_i = \sqrt{\dot{X}_i^2 + \dot{Y}_i^2 + \dot{Z}_i^2}$$

T	IG	MG	OG	VV	HV	LV
1.0000,	.87500,	-3.2500,	24.625,	897.00,	-1308.0,	-70.500,
2.0000,	.87500,	-3.1250,	24.625,	369.00,	-1332.5,	-58.500,
3.0000,	1.0000,	-3.1250,	24.750,	944.50,	-1420.5,	-66.000,
4.0000,	1.0000,	-3.2500,	24.875,	915.50,	-1474.5,	-64.000,
5.0000,	1.0000,	-3.2500,	25.000,	792.00,	-1530.5,	-61.500,
6.0000,	1.1250,	-3.2500,	24.875,	767.50,	-1588.0,	-59.000,
7.0000,	1.0000,	-3.1250,	25.000,	742.50,	-1645.5,	-57.500,
8.0000,	.87500,	-3.1250,	25.000,	715.50,	-1699.5,	-55.000,
9.0000,	1.0000,	-3.1250,	25.000,	686.50,	-1757.0,	-52.500,
10.000,	1.1250,	-3.1250,	25.000,	664.50,	-1817.0,	-50.500,
11.000,	1.2500,	-3.2500,	24.875,	634.00,	-1869.5,	-48.000,
12.000,	1.2500,	-3.2500,	24.875,	613.50,	-1928.0,	-40.000,
13.000,	1.1250,	-3.1250,	24.875,	587.00,	-1982.5,	-115.50,
14.000,	-.52500,	-2.5000,	22.500,	613.00,	-2058.5,	-107.50,
15.000,	1.3750,	-3.1250,	22.875,	587.00,	-2110.5,	-109.00,
16.000,	3.3750,	-4.1250,	24.250,	567.50,	-2190.5,	-121.00,
17.000,	4.0000,	-4.6250,	24.750,	534.00,	-2254.5,	-110.00,
18.000,	3.8750,	-4.6250,	24.375,	504.50,	-2320.5,	-107.00,
19.000,	3.8750,	-4.6250,	24.875,	487.50,	-2390.0,	-94.000,
20.000,	7.0000,	-5.1250,	24.375,	459.00,	-2454.5,	-75.500,
21.000,	14.500,	-8.6250,	22.000,	434.50,	-2503.5,	-51.500,
22.000,	25.750,	-3.2500,	18.875,	420.00,	-2559.5,	-20.000,
23.000,	40.625,	-12.750,	14.000,	411.00,	-2594.5,	6.5000,
24.000,	58.375,	-17.000,	3.0000,	405.00,	-2630.5,	44.500,
25.000,	69.375,	-21.125,	4.0000,	392.50,	-2669.5,	97.500,
26.000,	77.250,	-24.625,	2.1250,	386.50,	-2704.0,	160.50,
27.000,	80.500,	-26.000,	1.0000,	383.50,	-2735.0,	229.50,
28.000,	80.375,	-25.375,	1.0000,	376.00,	-2752.5,	300.00,
29.000,	80.000,	-25.000,	1.3750,	380.00,	-2771.0,	371.00,
30.000,	79.875,	-24.250,	1.6250,	373.00,	-2787.5,	447.00,
31.000,	79.875,	-24.125,	1.6250,	375.50,	-2801.0,	515.00,
32.000,	79.875,	-24.125,	1.5250,	373.50,	-2820.0,	587.50,
33.000,	79.875,	-24.250,	1.6250,	382.00,	-2838.5,	659.50,
34.000,	79.875,	-24.500,	1.5250,	380.50,	-2855.5,	730.00,
35.000,	79.875,	-24.500,	1.6250,	376.00,	-2872.5,	801.50,
36.000,	79.875,	-24.375,	1.6250,	375.00,	-2887.5,	877.50,
37.000,	80.000,	-24.375,	1.6250,	375.50,	-2900.5,	945.00,
38.000,	79.875,	-24.250,	1.6250,	370.00,	-2918.5,	1017.0,
39.000,	79.750,	-24.250,	1.6250,	373.00,	-2939.0,	1088.5,
40.000,	79.750,	-24.250,	1.6250,	375.00,	-2951.5,	1160.5,
41.000,	79.750,	-24.250,	1.7500,	372.50,	-2972.0,	1231.5,

Figure 8. Algorithm #1 Input Data

$$\dot{\phi}_i = TAN^{-1}(-\dot{Y}_i/\dot{X}_i), \text{ in proper quadrant}$$

$$\dot{\gamma} = TAN^{-1}\left(\sqrt{\dot{X}_i^{\,2}+\dot{Y}_i^{\,2}}/\dot{Z}_i\right), \text{ in proper quadrant}$$

Replacement with the following values will yield the acceleration components when used in the above formulae:

let FT = Frame Time

$SC_i = (T_i - T_{i-1}) \cdot FT$

then $VV_i = (VV_i - VV_{i-1})/SC_i$

$HV_i = (HV_i - HV_{i-1})/SC_i$

$LV_i = (LV_i - LV_{i-1})/SC_i$

4.1.2 Algorithm No. 1 - Comparison

- Input - Both approaches utilize the same data input for the test case (see Figure 8). Inherent in the conventional approach, however, is a capability of selectively ignoring certain input values (see Appendix B listing) which have been flagged as bad measurements. In order to duplicate this particular capability in the IPAA approach, the data file would have to have these records deleted.

- Implementation - A simple replacement of three velocity variables in the conventional approach allow the same basic algorithm to be utilized for acceleration computations. In addition, subroutine QUAD (see Appendix B) is utilized for four of the 12 computed values.

  The limitation on the number of blocks available in a diagram (180) forces the IPAA user to build two separate diagrams (see Appendix B). Also, the functional blocks representing the QUAD subroutine must be duplicated each time a conventional approach would invoke it.

- Execution - Scaling variables which may be specified during execution in the conventional approach entail a diagram modification of constant blocks prior to execution in the IPAA approach (e.g., Frame Time).

  Differences in execution time between the two approaches is not noticeable in the Univac time-sharing environment.

- Output - The output of both approaches for the test case are essentially the same (see Figures 9 and 10). Note, however, that due to the limited format associated with DISP blocks in the IPAA System, two of the output variables ($\dot{X}$,$\dot{Y}$) had to be scaled in order to print meaningful values.

  The computations for time 1.000 are not performed in the conventional approach, but must be in the IPAA approach in order that succeeding acceleration values be correct.

4.2 Algorithm Number Two

This algorithm is used in the analysis of rocket engine performance characteristics. Specifically, the algorithm establishes the relationship between the engine (overall) propellant mixture ratio and the gas generator (for turbopump power) propellant mixture ratio. The entire amount of one of the propellants, fuel or oxidizer, is supplied to the gas generator. The gas-generator portion of the remaining propellant constituent is also calculated.

4.2.1 Algorithm No. 2 - Mathematical Formulation

Let

$$T_i = \text{TIME}$$
$$EWF = 1/(T_i+1)$$
$$EWO = T_i \cdot EWF$$

OUTPUT VARIABLE (COLUMN) IDENTIFICATION:
1-PSEUDO TIME
2,3,4-VEHICLE XDOT, YDOT, ZDOT
5-TOTAL VELOCITY MAGNITUDE
6-ANGLE OF TOTAL VELOCITY VECTOR RELATIVE TO (+ LEFT OF) VEHICLE VERTICAL LONGITUDINAL PLANE (PHI)
7-ANGLE OF TOTAL VELOCITY VECTOR RELATIVE TO VEHICLE HORIZONTAL PLANE (GAMMA)
8-13 SAME AS 2-7 EXCEPT FOR ACCELERATION

VELOCITY SCALE FACTOR IS 1.000 CTS/METER/SEC. FRAME TIME IS .500000 SEC.

2.000	-873.9	-41.79	1360.	1617.	177.3	57.3	-122.5	-.7213	-5.509	122.6	179.7	-2.6
3.000	-933.7	-44.61	1364.	1654.	177.3	55.6	-126.0	.2991	4.014	126.0	-179.9	1.8
4.000	-991.5	-43.29	1364.	1687.	177.5	54.0	-121.6	-1.064	-5.380	121.7	179.5	-2.5
5.000	-1050.	-43.16	1367.	1724.	177.6	52.5	-122.3	.2617	2.869	122.4	-179.9	1.3
6.000	-1115.	-47.51	1367.	1765.	177.6	50.8	-125.0	-2.038	3.968	125.0	173.1	1.8
7.000	-1175.	-45.60	1371.	1806.	177.8	49.4	-125.5	.2620	3.233	125.5	-179.9	1.5
8.000	-1236.	-41.82	1369.	1845.	178.1	47.9	-120.8	.4014	-3.364	120.9	-179.8	-1.6
9.000	-1300.	-45.61	1367.	1887.	178.0	46.4	-129.8	-.1741	-4.005	128.9	173.9	-1.8
10.000	-1363.	-49.81	1373.	1935.	177.9	45.2	-127.4	-.7579	10.84	127.9	179.7	4.9
11.000	-1425.	-52.66	1364.	1974.	177.9	43.7	-119.2	-.7108	-12.03	113.9	179.7	-5.8
12.000	-1488.	-53.13	1371.	2024.	178.0	42.6	-125.3	-.9236	12.87	125.9	179.6	5.9
13.000	-1552.	-122.3	1376.	2078.	175.5	41.5	-127.7	-144.0	10.47	192.8	131.6	3.1
14.000	-1665.	-58.00	1362.	2152.	178.0	3 .3	-104.3	20.15	108.3	151.7	-169.1	45.6
15.000	-1716.	-127.5	1370.	2139.	175.7	3 .5	-130.7	-9.164	-11.70	131.6	176.0	-5.1
16.000	-1748.	-198.3	1427.	2266.	173.5	39.0	-150.3	-15.48	25.95	153.1	174.1	9.8
17.000	-1904.	-223.1	1441.	2319.	172.9	38.4	-143.8	-12.31	-6.623	144.5	175.1	-2.6
18.000	-1873.	-222.0	1446.	2377.	173.2	37.5	-144.5	-7.682	2.431	144.7	177.0	1.0
19.000	-1950.	-215.6	1462.	2447.	173.7	36.7	-153.1	12.34	32.21	156.9	-175.2	11.9
20.000	-2009.	-331.7	1448.	2498.	170.6	35.4	-135.0	17.11	-5.566	136.2	-172.8	-2.3
21.000	-2038.	-623.6	1395.	2547.	163.0	33.2	-129.6	13.27	-.7142	129.3	-174.1	-.3
22.000	-1979.	-1050.	1307.	2594.	152.1	30.3	-118.9	6.070	9.805	119.4	-177.6	4.7
23.000	-1728.	-1552.	1229.	2627.	139.1	27.9	-91.21	-5.789	7.135	91.69	176.4	4.5
24.000	-1316.	-1953.	1242.	2662.	124.0	27.8	-103.2	-21.18	8.073	105.6	168.4	4.4
25.000	-927.1	-2157.	1333.	2700.	113.3	29.6	-127.1	-42.28	-.2182	134.0	161.6	-.1
26.000	-699.6	-2204.	1462.	2736.	107.6	32.3	-137.8	-40.90	10.66	144.2	163.5	4.2
27.000	-651.3	-2222.	1532.	2771.	106.3	33.3	-146.1	-37.12	13.98	151.4	165.7	5.3
28.000	-729.7	-2232.	1514.	2794.	108.1	32.8	-145.0	-15.95	-5.296	146.0	173.7	-2.1
29.000	-811.5	-2254.	1430.	2821.	109.8	31.9	-140.9	-7.560	14.78	147.9	177.1	5.7
30.000	-884.9	-2275.	1467.	2848.	111.3	31.0	-146.5	-7.110	6.262	146.8	177.2	2.4
31.000	-958.4	-2290.	1450.	2873.	112.8	30.6	-146.6	-4.424	-2.627	146.7	178.3	-1.0
32.000	-1032.	-2285.	1468.	2905.	114.3	30.3	-147.1	-8.745	14.87	148.1	176.6	5.7
33.000	-1106.	-2295.	1480.	2939.	115.9	30.2	-147.9	-7.254	15.15	149.8	177.2	5.9
34.000	-1179.	-2287.	1493.	2972.	117.3	30.2	-146.7	-8.826	4.986	147.0	176.6	1.9
35.000	-1252.	-2299.	1492.	3006.	118.7	29.8	-146.8	-11.31	-.5712	147.3	175.6	-.2
36.000	-1326.	-2294.	1490.	3040.	120.0	29.4	-146.9	-3.842	5.910	147.1	178.5	2.3
37.000	-1393.	-2300.	1474.	3076.	121.2	29.1	-146.3	-6.499	5.997	146.5	177.5	2.3
38.000	-1473.	-2304.	1457.	3113.	122.6	26.5	-147.2	-7.953	-3.213	147.4	176.9	-1.2
39.000	-1552.	-2307.	1493.	3156.	123.9	28.2	-147.7	-11.55	14.37	148.9	175.5	5.7
40.000	-1625.	-2306.	1497.	3194.	125.2	27.9	-146.0	2.575	7.370	146.2	-179.0	2.9
41.000	-1697.	-2319.	1503.	3233.	126.3	27.7	-148.9	-15.49	6.037	149.9	174.1	2.3

Figure 9. Algorithm #1 Output (Conventional)

IPAA ANALYSIS ROUTINE

OUTPUT OF THE FIRST COMPARATIVE ALGORITHM (VELOCITY SECTION)

EXECUTION RESULTS

TIME	$\dot{x}$ 903	$\dot{y}$ 910	$\dot{z}$ 912	mag 914	917	919
1.000	-81.263	-3.947	1363.275	1587.590	177.219	59.172
2.000	-87.391	-4.179	1362.440	1617.486	177.262	57.255
3.000	-93.367	-4.461	1364.414	1653.891	177.265	55.586
4.000	-99.145	-4.329	1363.544	1636.689	177.500	53.959
5.000	-104.963	-4.316	1367.439	1724.377	177.645	52.467
6.000	-111.483	-4.751	1367.219	1764.766	177.566	50.781
7.000	-117.488	-4.560	1371.031	1806.179	177.777	49.385
8.000	-123.553	-4.182	1369.299	1844.794	178.061	47.923
9.000	-129.970	-4.561	1367.404	1887.084	177.990	46.437
10.000	-136.315	-4.981	1372.934	1935.356	177.907	45.156
11.000	-142.543	-5.266	1364.161	1973.715	177.884	43.722
12.000	-148.806	-5.313	1370.597	2023.779	177.956	42.629
13.000	-155.222	-12.228	1375.595	2077.645	175.496	41.460
14.000	-166.503	-5.600	1361.907	2151.963	178.005	39.265
15.000	-171.585	-12.752	1369.532	2199.096	175.750	38.519
16.000	-174.823	-19.832	1427.245	2265.539	173.528	39.049
17.000	-180.360	-22.307	1441.278	2319.488	172.949	38.417
18.000	-187.339	-22.204	1446.310	2377.118	173.241	37.476
19.000	-194.992	-21.557	1462.417	2446.898	173.691	36.703
20.000	-200.861	-33.166	1447.586	2498.006	170.624	35.415
21.000	-203.780	-62.359	1395.394	2547.273	162.985	33.216
22.000	-197.927	-104.965	1307.085	2593.824	152.065	30.260
23.000	-172.766	-155.190	1228.011	[illegible]	138.008	27.869
24.000	-131.567	-195.268	1241.603	2661.867	123.971	27.803
25.000	-92.709	-215.700	1333.276	2699.862	113.858	29.501
26.000	-69.966	-220.423	1462.460	2738.194	107.608	32.309
27.000	-65.129	-222.233	1522.184	2771.275	106.334	33.317
28.000	-72.968	-223.226	1514.138	2794.281	108.101	32.811
29.000	-81.161	-225.416	1490.320	2821.498	109.799	31.884
30.000	-88.489	-227.508	1466.761	2847.881	111.253	31.000
31.000	-96.835	-228.043	1460.483	2872.599	112.796	30.569
32.000	-103.190	-228.481	1467.817	2906.107	114.306	30.348
33.000	-110.571	-228.520	1480.368	2938.814	115.819	30.247
34.000	-117.876	-228.336	1492.765	2971.794	117.304	30.153
35.000	-125.217	-228.902	1492.480	3006.833	118.680	29.771
36.000	-132.577	-229.411	1490.439	3040.071	120.024	29.358
37.000	-139.301	-230.007	1494.500	3076.416	121.201	29.064
38.000	-147.265	-230.438	1486.753	3112.748	122.582	28.531
39.000	-153.239	-230.710	1492.996	3156.813	123.936	28.231
40.000	-162.539	-230.582	1486.681	3193.646	125.180	27.947
41.000	-159.656	-231.366	1503.248	3238.919	126.263	27.653

END OF ANALYSIS

Figure 10. Algorithm #1 Output (IPAA)

IPAA ANALYSIS ROUTINE

ALGORITHM NUMBER ONE ANALYSIS-ACCELERATION SECTION

EXECUTION RESULTS

TIME	902	905	912	914	917	919
1.000	********	-78.935	2726.550	3175.181	177.219	59.172
2.000	-122.483	-.721	-5.509	122.609	179.662	-2.575
3.000	-125.960	.299	4.014	126.024	-179.864	1.825
4.000	-121.597	-1.064	-5.380	121.721	179.499	-2.533
5.000	-122.312	.262	2.869	122.352	-179.877	1.344
6.000	-124.960	-2.038	3.968	125.040	179.066	1.819
7.000	-125.457	.262	3.235	125.499	-179.880	1.477
8.000	-120.804	.401	-3.364	120.851	-179.810	-1.595
9.000	-128.833	-.174	-4.006	128.895	179.923	-1.781
10.000	-127.413	-.758	10.838	127.875	179.659	4.862
11.000	-119.205	-.711	-12.632	119.812	179.658	-5.764
12.000	-125.266	-.924	12.872	125.929	179.578	5.867
13.000	-127.690	-144.013	10.468	192.754	131.562	3.113
14.000	-104.297	20.149	103.338	151.727	-182.066	45.564
15.000	-130.722	-9.154	-11.698	131.564	175.994	-5.101
16.000	-150.093	-15.478	25.951	153.105	174.112	9.759
17.000	-143.811	-12.314	-6.623	144.489	175.106	-2.627
18.000	-144.436	-7.682	2.431	144.718	176.956	.962
19.000	-153.061	12.943	32.214	156.949	-175.166	11.844
20.000	-134.965	17.105	-5.566	136.158	-172.777	-2.343
21.000	-128.575	13.265	-.714	129.259	-174.110	-.317
22.000	-118.925	5.070	9.806	119.436	-177.559	4.709
23.000	-91.213	-5.789	7.185	91.679	176.368	4.495
24.000	-103.161	-21.180	8.073	105.622	168.398	4.384
25.000	-127.113	-42.276	-.218	133.959	161.604	-.093
26.000	-137.821	-40.899	10.664	144.156	163.471	4.242
27.000	-146.119	-37.120	13.981	151.407	165.746	5.298
28.000	-144.982	-15.946	-5.296	145.962	173.723	-2.079
29.000	-146.940	-7.561	14.779	147.875	177.054	5.736
30.000	-146.456	-7.110	6.262	146.762	177.221	2.445
31.000	-146.637	-4.424	-2.627	146.728	178.272	-1.026
32.000	-147.096	-8.745	14.668	148.084	176.597	5.685
33.000	-147.892	-7.254	15.145	148.842	177.192	5.840
34.000	-146.670	-8.826	4.226	147.017	176.556	1.905
35.000	-146.225	-11.314	-.571	147.262	175.594	-.222
36.000	-146.923	-3.842	5.910	147.092	178.502	2.303
37.000	-146.273	-6.499	5.997	146.540	177.456	2.346
38.000	-147.182	-7.953	-3.213	147.431	178.907	-1.249
39.000	-147.651	-11.532	14.871	148.846	175.534	5.734
40.000	-146.000	2.575	7.370	146.209	-178.989	2.889
41.000	-148.883	-15.478	6.037	149.807	174.065	2.310

END OF ANALYSIS

Figure 10. Algorithm #1 Output (IPAA)(Continued)

PBWF = .75-EWO

OPBMR = EWO/PBWF

PCTPBF = PBWF/EWF

PBWO = .75-EWF

FPBMR = PBWO/EWF

PCTPBO = PBWO/EWO

RPBMR = 1/FPBMR

4.2.2 Algorithm No. 2 - Comparison

- Input - No input required for this algorithm.
- Implementation - The implementation of this algorithm is straightforward whether the conventional or IPAA approach is taken.

  This particular algorithm, however, had the approaches implemented in a parallel fashion. The one feature that stood out was the ability of an analyst unfamiliar with the Operating System to readily implement the IPAA approach.
- Output - The conventional output has the advantage of providing a column heading to identify the variable on the printout (see Figure 11). The IPAA output has the feature of isolating when and where (block number) an error occurred (see Figure 12).

```
ENTER START, STOP, INCREMENT
> 0.0, 3.0, 0.1
   EMMR          OPBMMR         PCTPBF         FPBMMR         RPBMMR         PCTPBO
 .00000000      .00000000      .75000000+00  -.25000000+00  -.40000000+01   .00000000
 .10000000+00   .13793103+00   .72499999+00  -.17500000+00  -.57142856+01  -.17500000+01
 .20000000+00   .28571425+00   .70000000+00  -.10000000+00  -.99999996+01  -.50000001+00
 .30000000+00   .44444443+00   .67500000+00  -.24999999-01  -.40000002+02  -.83333329-01
 .39999999+00   .61538459+00   .65000001+00   .49999997-01   .20000001+02   .12499999+00
 .49999999+00   .79999997+00   .62500000+00   .12499999+00   .80000005+01   .24999999+00
 .59999999+00   .99999996+00   .60000000+00   .19999998+00   .50000004+01   .33333332+00
 .69999995+00   .12173912+01   .57500003+00   .27499999+00   .36363637+01   .39285715+00
 .79999997+00   .14545453+01   .55000001+00   .34999999+00   .28571430+01   .43750000+00
 .89999997+00   .17142855+01   .52500003+00   .42499999+00   .23529412+01   .47222223+00
 .99999996+00   .19999998+01   .50000001+00   .49999998+00   .20000001+01   .50000000+00
 .10999999+01   .23157892+01   .47500002+00   .57499997+00   .17391305+01   .52272727+00
 .11999999+01   .26666664+01   .45000001+00   .64999995+00   .15384617+01   .54166665+00
 .12999999+01   .30563232+01   .42500002+00   .72499996+00   .13793104+01   .55769230+00
 .13999999+01   .34999996+01   .40000002+00   .79999994+00   .12500001+01   .57142856+00
 .14999999+01   .39999996+01   .37500002+00   .87499993+00   .11428572+01   .58333332+00
 .15999999+01   .45714279+01   .35000003+00   .94999995+00   .10526316+01   .59375000+00
 .16999999+01   .52307683+01   .32500004+00   .10249999+01   .97560982+00   .60294117+00
 .17999999+01   .59999989+01   .30000003+00   .10999999+01   .90909096+00   .61111111+00
 .18999999+01   .69090891+01   .27500005+00   .11749999+01   .85106387+00   .61842106+00
 .19999999+01   .79999981+01   .25000004+00   .12499999+01   .80000004+00   .62500000+00
 .20999999+01   .93333304+01   .22500005+00   .13249999+01   .75471704+00   .63095238+00
 .21999999+01   .10999996+02   .20000006+00   .13999999+01   .71428575+00   .63636364+00
 .22999999+01   .13142851+02   .17500007+00   .14749999+01   .67796613+00   .64130435+00
 .23999999+01   .15999992+02   .15000006+00   .15499999+01   .64516133+00   .64583333+00
 .24999999+01   .19999983+02   .12500007+00   .16249999+01   .61538464+00   .65000001+00
 .25999998+01   .25999980+02   .10000007+00   .16999999+01   .58823532+00   .65384616+00
 .26999998+01   .35999968+02   .75000063-01   .17749999+01   .56338032+00   .65740740+00
 .27999998+01   .55999895+02   .50000090-01   .18499999+01   .54054057+00   .66071429+00
 .28999998+01   .11599973+03   .25000056-01   .19249999+01   .51948055+00   .66379310+00
 .29999998+01   .50331647+08   .59604642-07   .19999999+01   .50000003+00   .66666666+00
*************************************************************
*                     ARITHMETIC FAULT SUMMARY              *
*                                                           *
*    WARNING - AT LEAST ONE DIVIDE CHECK HAS OCCURRED I[illegible]   *
*                        PROGRAM EXECUTION                  *
*************************************************************
```

Figure 11. Algorithm #2 Output (Conventional)

IPAA ANALYSIS ROUTINE
EXECUTION RESULTS

TIME EMMR	204 OPBMR	209 PCTPBF	305 RPBMR	309 FPBMR	314 PCTPBO
ATTEMPTED DIVISION BY ZERO - BLOCK 313					
.000	.000	.750	-4.000	-.250	.000
.100	.138	.725	-5.714	-.175	-1.750
.200	.256	.700	-10.000	-.100	-.500
.300	.444	.675	-40.000	-.025	-.083
.400	.615	.650	20.000	.050	.125
.500	.800	.625	8.000	.125	.250
.600	1.000	.600	5.000	.200	.333
.700	1.217	.575	3.636	.275	.393
.800	1.455	.550	2.857	.350	.438
.900	1.714	.525	2.353	.425	.472
1.000	2.000	.500	2.000	.500	.500
1.100	2.316	.475	1.739	.575	.523
1.200	2.667	.450	1.538	.650	.542
1.300	3.059	.425	1.379	.725	.558
1.400	3.500	.400	1.250	.800	.571
1.500	4.000	.375	1.143	.875	.583
1.600	4.571	.350	1.053	.950	.594
1.700	5.231	.325	.976	1.025	.603
1.800	6.000	.300	.909	1.100	.611
1.900	6.909	.275	.851	1.175	.618
2.000	8.000	.250	.800	1.250	.625
2.100	9.333	.225	.755	1.325	.631
2.200	11.000	.200	.714	1.400	.636
2.300	13.143	.175	.678	1.475	.641
2.400	16.000	.150	.645	1.550	.646
2.500	20.000	.125	.615	1.625	.650
2.600	26.000	.100	.588	1.700	.654
2.700	36.000	.075	.563	1.775	.657
2.800	56.000	.050	.541	1.850	.661
2.900	116.000	.025	.519	1.925	.664
3.000	********	000	.500	2.000	.667

Figure 12. Algorithm #2 Output (IPAA)

5.0 CONCLUSIONS

A sophisticated interactive analysis aid has been successfully installed on the Univac 1110. This aid is capable of significantly enhancing the analyst's response by provision of a highly interactive menu for (a) modeling new analysis problems as diagrams, (b) executing the analysis diagrams, and (c) evaluating the execution results.

Analysis diagrams can be created, edited, modified, stored, retrieved, annotated, and executed at the analyst's graphics station. The diagrams themselves are mathematical in nature to provide the widest flexibility and applicability. Available special features include handling multi-page diagrams, parallel analysis processes, nonlinear analysis and automated input and output of sampled data.

The recommended development of macros within the software system will allow system modeling at a modular or functional level as opposed to the current primitive level.

6.0 RECOMMENDATIONS

The following subsections describe recommended modifications/additions to the IPAA software system. Some are logical extensions of the software design, while others are the result of analyst interviews, feedback from user classes, and general discussions with the customer.

Before pursuing the individual, specific recommendations, one over-all suggestion is presented. Interactive software systems, besides possessing good system design which lends itself to modification and expansion, must go through a period of "operator-proofing." Ideally, an interactive software system will not allow the analyst to "bomb" during its execution due to an operator-error. This ideal is seldom met in practice, however, due primarily to the inability of the software system to determine the intent of the analyst at all times, and secondarily the inability of the analyst to communicate directly with the software through the terminal, e.g., the bi-directional communication of the analyst and software system generally passes through a system device handler which may intercept certain commands and attempt executive system functions.

Additionally, the initial implementation of an interactive system is rarely entirely pleasing to the user. Even assuming the software system will perform all the functions the analyst desires, the method of implementation may make its use less desirable. As an example, consider the entry of parameters in the IPAA system. The designer may consider the following alternatives:

1. Have the user enter all parameters.
2. Have the user enter each parameter until a carriage return only is detected. This signals end of input and the remaining parameters are to remain unchanged.
3. Have the user enter carriage return only if he wishes the parameter to remain unchanged. Otherwise enter the new parameter. This method forces the user to exhaust the list of parameters, whether they are changed or not.

4. Have the operator enter the parameter number of the parameter he wishes to change.

Even through any of the methods will accomplish the task, the method chosen by the software designer may not be as desirable as another method for the user. Frequently this will not be discovered until actual use on real problems by the analyst. Many times these "fielder's choice" options do not require extensive re-design of the software system, only fairly straightforward modification. Should they have a serious effect on the software design, a suitable compromise can usually be worked out. Therefore, in the interest of providing useful, well-designed, and error-free software, it is recommended that future software modifications of the IPAA software system be implemented in an incremental manner, i.e., as each module or modification is completed, this latest version of IPAA be turned over to a select group of analysts for testing, and, if necessary, recommendations for alteration.

6.1 Macro Capability

The IPAA software system was designed with the eventual inclusion of macros in mind. To the analyst, it means modeling at a system level as opposed to the present primitive level.

6.1.1 Macro Definition

A macro, in IPAA terms, is a method of referencing a previously created diagram by the use of a single functional block. As an example, the page representing the QUAD subroutine in the Algorithm No. One diagram (diagram page 6) would be referenced by a block with the name QUAD for function.

6.1.2 Macro Implementation

The macro capability must have certain constraints in order for it to work within the present framework of the IPAA system.

A top-level view of a macro implementation/modification to the IPAA software system may be best understood by presenting an example and discussing the design considerations involved.

Figure 13 depicts the QUAD subroutine in diagram form. This particular "page" may be created with the current IPAA system; however, a method of storing this "page" in a library must be developed. Inclusion of the block depicted in Figure 14 within an IPAA diagram tells us that a method of replacing that block prior to execution with the stored "page" must be developed. From this simple macro example, the following design criteria may be inferred:

1. The maximum length of a macro must be established. It is reasonable to allow the full nine pages.

2. The not-so-obvious constraints; certain current blocks will not be allowed in macros, e.g., input and output blocks except as page connectors.

3. The maximum number of inputs to a macro will be four, for compatibility with the current software. (In the example there are two.)

4. Although this example would not require parameter modification prior to execution, it is possible that a method to do so would be desirable for other macros.

In addition, the available resources of the computer system will have an effect on the design. For example, the maximum number of blocks allowed for any execution (both diagram and macros) would be constrained by the amount of memory available at execution time. The method of storing and recalling a library of macros will be defined in large part by the capabilities and peculiarities of the operating system.

In conclusion, it may be stated that the inclusion of a macroing capability in the IPAA software is not only feasible, but may be implemented in a number of ways.

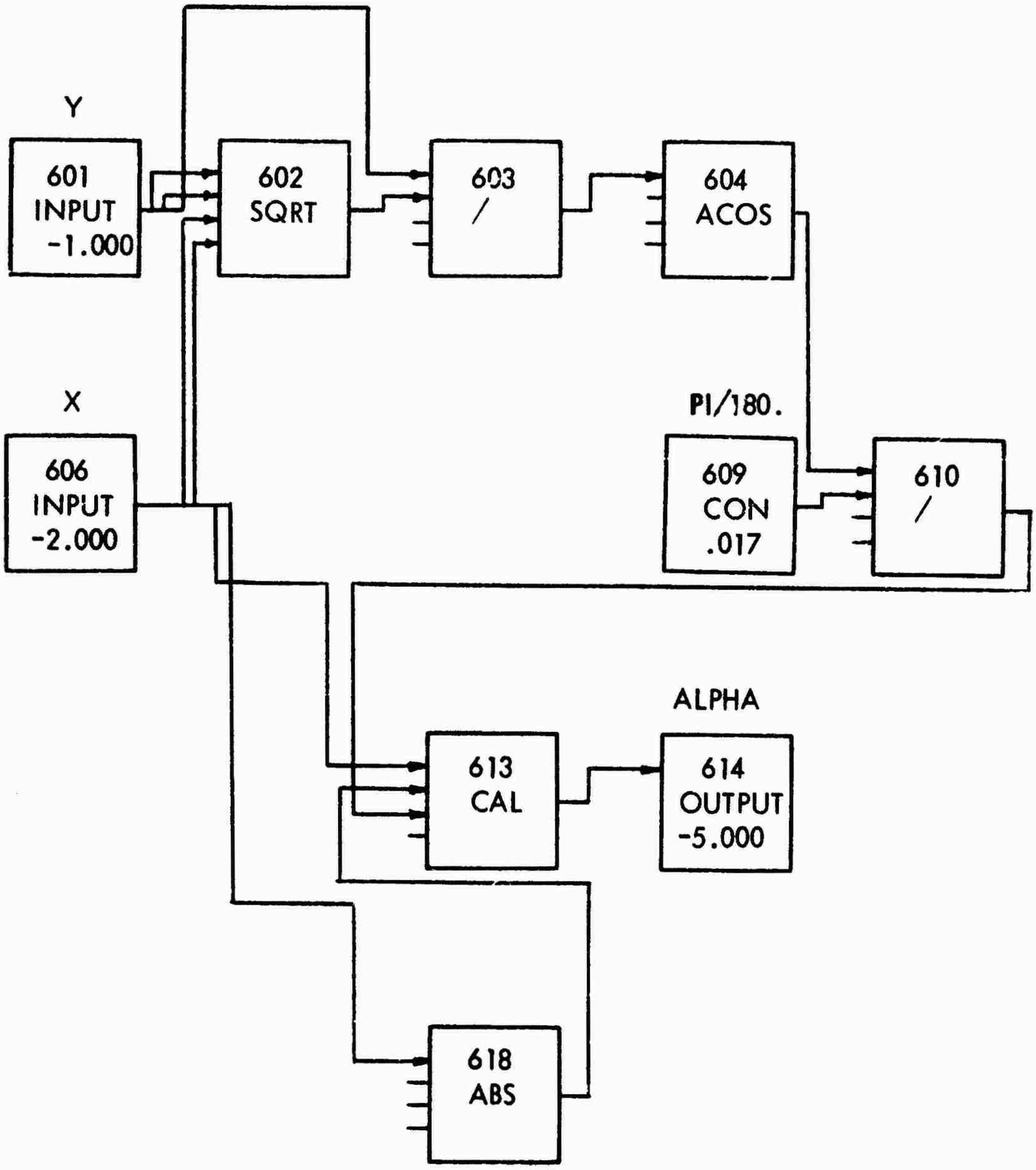


Figure 13. Quad Macro

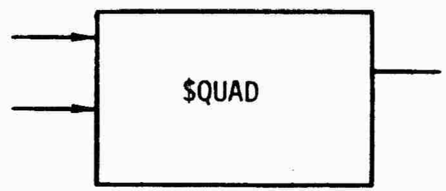


Figure 14. QUAD MACRO REFERENCE

6.2 Graphic Display of Output Data

Analysis of processed data is less error prone and faster when displayed to the analyst in a graphic form as opposed to a columnar tabulation.

Inherent in a discussion of interactive displays is the capability for the analyst to expand displays both horizontally and vertically in order to view portions of data in more detail.

Experience has shown that many times analysts who have interactive displays available will develop measurement techniques which they wish to be included as options in the interactive display. The proper design of an interactive display will make possible its evolution into an interactive analysis position. As an example, an analyst may wish to measure the time between two events on his display. He should be able to simply mark the events with his cursor and get a rapid printout of this time difference.

6.3 Current System Modifications

Several suggestions have been made by analysts for modifications to the current IPAA software.

6.3.1 Uniscope Execution

It is possible to modify the software so that execution from a Uniscope terminal is possible. There was more interest in the initial stages of the contract for this capability than there was toward the end, so discussions with the analysts may show this to be of marginal value.

6.3.2 Addition of Functional Blocks

The capability of adding functions to the IPAA system exists and will continue whether or not a macroing capability is introduced. Assuming there was a macroing capability on the IPAA system, there would still exist a need for additional functions, since all models do not lend themselves to macroing.

Specific requests are:

- Random number generator block.
- Expansion of FGEN1 capability to allow storage of up to 200 table pair values.
- Labeling capability for the tabular output generated by DISP blocks.
- Capability to produce a greater range of values in the tabular output generated by DISP blocks.

7.0 REFERENCES

1. James B. Scarborough, Numerical Mathematical Analysis, John Hopkins Press, Sixth Edition, 1966.

2. B. Carnahan, H. A. Luther, and J. O. Wilkes, Applied Numerical Methods, John Wiley and Sons, New York, 1969.

3. User's Manual, Interactive Programming and Analysis Aid, Harris Electronic Systems Division, Melbourne, FL, April 1978.

APPENDIX A

UTILIZATION OF LINES AS BLOCK CONNECTORS

A discussion of an algorithmic approach to the problem of selectively connecting displayed blocks without superimposition of lines is presented.

Consider Figure 1. A user area is depicted having a possible 20 blocks which may be defined (shaded areas). (These shaded areas, whether defined or not, will not have lines drawn through them.)

It is helpful to view the shaded blocks as a 4x5 array, while the entire user area may be viewed as a 9x11 array.

In order to implement this on a digital computer, an array of length 9x11x2 may be set up in core, i.e., there are two words representing each block in the 9x11 array. One word will be a counter for the number of horizontal lines existing within that space, while the other will be a counter for the number of vertical lines in that space.

Now, if this array is initialized to zero and the words representing the shaded blocks are flagged so they won't be used, the algorithm may be defined.

Definition of variables used:

DX = distance between vertical lines

DY = distance between horizontal lines

R_1, C_1 = output block location in array

R_2, C_2 = input block location in array

Note that all outputs are from the right while inputs are into the left of the respective blocks.

The determination of path is, of course, dependent on the location of the input block relative to the output block (above, below, left, right) in the array.

In general, however, a vertical line is drawn one DX greater than the maximum of the vertical line counter for the spaces it will pass through, and similarly for the horizontal lines.

Figure 1 shows an example of two lines being drawn with this algorithm. Note that all lines proceed from output pin to input pin and all lines crossing should be ignored while tracing a path.

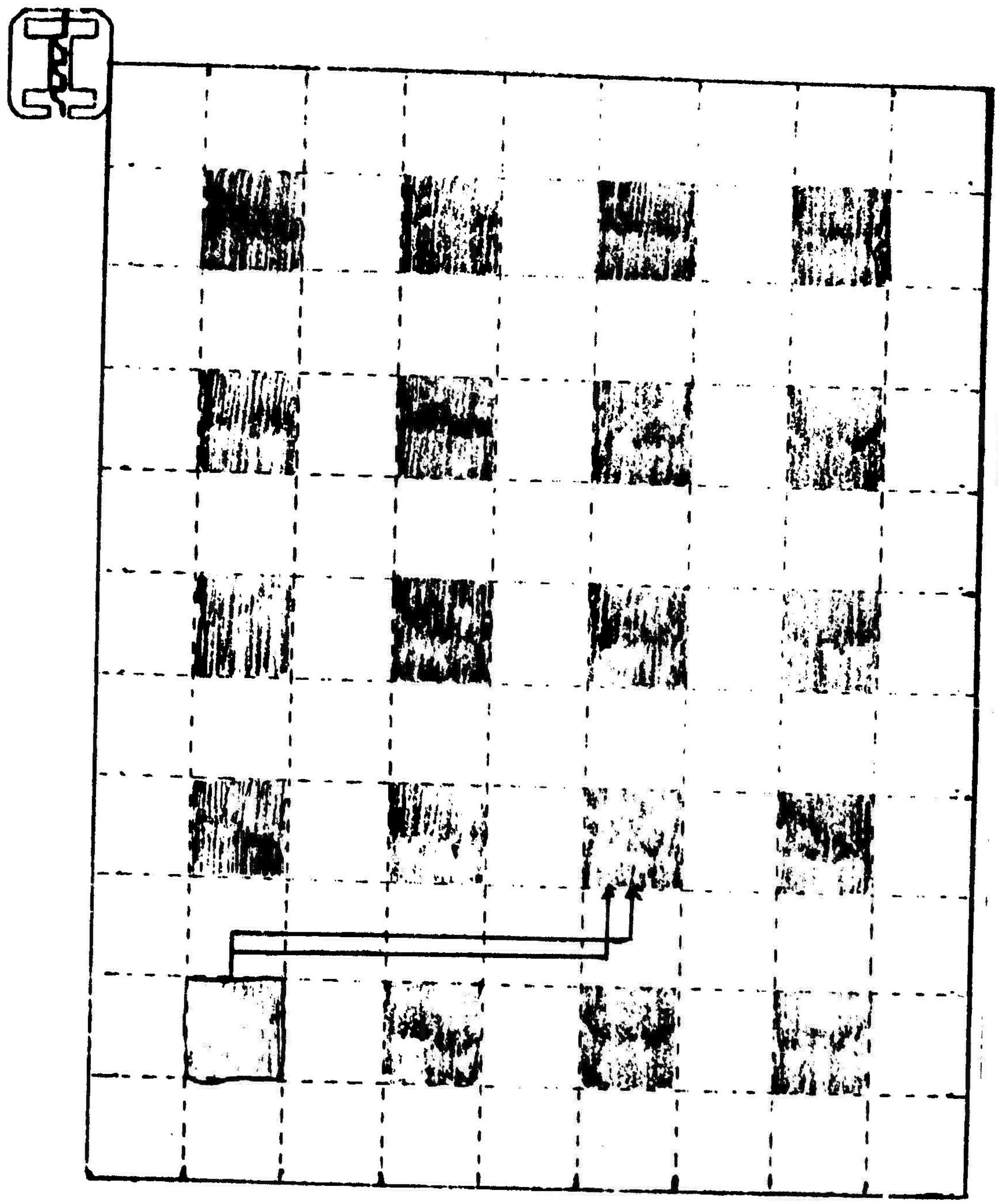


Figure 1.

APPENDIX B

ALGORITHMS LISTINGS AND DIAGRAMS

SECTION B.1.

ALGORITHM NO. 1 CONVENTIONAL CODE LISTING

```
MOC-SUPG*G(1).MAIN
          COMPILER(DIAG=3)
          WRITE(6,20)
          READ(5,10)IOPT
          GO TO (1,2,3,4,5,6),IOPT
    1     CALL VEHCLE
          STOP
    2     CALL VEL
          STOP
    3     CALL GIMBAL
          STOP
    4     CALL ANGLE
          STOP
    5     CALL SENSE
          STOP
    6     CALL MATH
          STOP
   10     FORMAT()
   20     FORMAT(/5X,'ENTER OPTION CODE:',/,
         *5X,'1 - VEHICLE',/,
         *5X,'2 - TOTAL VELOCITY',/,
         *5X,'3 - EULER ANGLE TRANSFORMATIONS',/,
         *5X,'4 - INSTRUMENT MOUNTING ANGLES',/,
         *5X,'5 - SENSED VELOCITY FROM INSTRUMENTS',/,
         *5X,'6 - MATH FUNCTIONS')
          END
```

```
NUF-SEG,G*C(1).VEHCLE
      COMPILER(DIAG=3)
      SUBROUTINE VEHCLE
      PARAMETER IPRA=12,IPRB=2500,IPRC=IPRA/2
      COMMON S(IPRA,IPRB),K,KOUNT
C
C     SEE FORMAT 1001 FOR PROGRAM DESCRIPTION
C
C     PROGRAM CALLS SUBROUTINES PBV,TMRD,IMSORT,QUAD
C
      K=6
      WRITE(K,1000)
      READ(5,11)CON
      IF(CON.EQ.'Y')WRITE(6,1001)
      CALL TMRD
      CALL PBV
      STOP
   10 FORMAT()
   11 FORMAT(A1)
 1000 FORMAT(5X,'NEED PROGRAM DESCRIPTION?')
 1001 FORMAT(///,10X,'VERT, HORIZONTAL AND LATERAL SENSED',/,
     *5X,'VELOCITY PROJECTED ONTO VEHICLE REFERENCE AXES. VEHICLE',/,
     *5X,'X, Y, Z COMPONENTS AND TOTAL VELOCITY ARE COMPUTED.',/,
     *5X,'SENSED ACCELERATIONS ARE COMPUTED SIMILARLY.  ENTER DATA',/,
     *5X,'FILES AS FOLLOWS:',/,
     *5X,'1-INNER GIMBAL COUNTS',/,
     *5X,'2-MIDDLE GIMBAL COUNTS',/,
     *5X,'3-OUTER GIMBAL COUNTS',/,
     *5X,'4-VERTICAL SENSED VELOCITY',/,
     *5X,'5-HORIZONTAL SENSED VELOCITY',/,
     *5X,'6-LATERAL SENSED VELOCITY',//)
      END
```

MDC-SCBG.G(1).TMRD

```
      COMPILER(DIAG=3)
      SUBROUTINE TMRD
      PARAMETER IPRA=12,IPRB=2500,IPRC=IPRA/2
      COMMON S(IPRA,IPRB),K,KOUNT
      COMMON /A/DT(IPRC)
      DIMENSION FMT(20),V(30),T(IPRC)
      DIMENSION CHAN1(IPRC),CHAN2(IPRC)
C
C     VALUE OF K (NO. OF FILES) MUST BE SET BY CALLING
C     PROGRAM.  SET K=0 IF NUMBER OF FILES IS TO BE
C     READ IN AT PROGRAM EXECUTION TIME.
C
      KOUNT1=0 @DATA POINT COUNT OF REFERENCE FILE
      ICODE=5
      EDIT='N'
      IF(K.NE.0)GO TO 3
      WRITE(6,101)
    4 CONTINUE
      READ(5,10)K
      IFILE=IPRC
      IF(K.GT.IFILE)WRITE(6,102)IFILE
      IF(K.GT.IFILE)GO TO 4
    3 CONTINUE
      FORMAT='Y'
      IF(K.GT.1)WRITE(6,103)
      IF(K.GT.1)READ(5,11)FORMAT
      DO 1 J1=1,K
      IF(J1.EQ.1.OR.FORMAT.NE.'Y')WRITE(6,104)
      IF(J1.EQ.1.OR.FORMAT.NE.'Y')READ(5,10)IFORM
      IF(J1.GT.1.AND.FORMAT.EQ.'Y')GO TO 5
      IF(IFORM.LT.3)WRITE(6,105)
      IF(IFORM.LT.3)READ(5,11)EDIT
      IF(EDIT.EQ.'Y')ICODE=0
      IF(IFORM.GT.2)WRITE(6,109)
      IF(IFORM.GT.2)READ(5,10)N
```

```
      IF(IFORM.EQ.3)WRITE(6,110)
      IF(IFORM.EQ.3)READ(5,100)(FMT(I),I=1,20)
      IF(IFORM.EQ.4)WRITE(6,111)
      IF(IFORM.EQ.4)READ(5,10)IF
    5 CONTINUE
      IF(IFORM.GT.2)WRITE(6,112)
      IF(IFORM.GT.2)READ(5,10)IX,IY
      J=0
      IF(IFORM.LT.4)WRITE(6,106)J1
      I1=2*J1-1
      I2=I1+1
   16 CONTINUE
      J=J+1
   17 CONTINUE
      IF(IFORM.EQ.1)READ(5,107,END=19,ERR=17)IEDIT,S(I1,J),
     *S(I2,J),R
      IF(IFORM.EQ.2)READ(5,108,END=19,ERR=17)S(I1,J),S(I2,J),
     *IR,IEDIT
      IF(IFORM.EQ.3)READ(5,FMT,END=19,ERR=17)(V(N2),N2=1,N)
      IF(IFORM.EQ.4)READ(IF,END=19,ERR=17)(V(N1),N1=1,N)
      IF(IFORM.GT.2)S(I1,J)=V(IX)
      IF(IFORM.GT.2)S(I2,J)=V(IY)
      IF(IFORM.LT.2)GO TO 20
      IF(IEDIT.GT.ICODE)GO TO 17
   20 CONTINUE
      IF(S(I2,J).LE.0)GO TO 17
      IF(S(I1,J).GT.99999999)GO TO 17
      IF(J.GT.1)GO TO 18
      IT=S(I1,J)
      AT=IT
      T(J1)=S(I1,J)-AT
      DT(J1)=T(J1)-T(1)
   18 CONTINUE
      GO TO 16
```

```
 9    CONTINUE
      KOUNT=J-1
      WRITE(6,115)J1,KOUNT
      IF(J1.EQ.1)KOUNT1=KOUNT
      IF(IFOR.EQ.4)REWIND IF
      CONTINUE
      KOUNT=KOUNT1
      CALL TMSORT

      PRINT OUT SORTED DATA

      WRITE(6,114)KOUNT,S(1,1),S(1,KOUNT)
      WRITE(6,123)
      READ(5,11)PRT
      IF(PRT.NE.'Y')GO TO 24
      IA=1
      IB=KOUNT
      IF(PRT.EQ.'Y')WRITE(6,115)KOUNT
      IF(PRT.EQ.'Y')READ(5,10)IA,IB
      IF(IA.EQ.0.OR.IA.LT.0)IA=1
      IF(IB.GT.KOUNT)IB=KOUNT
      IF(PRT.EQ.'Y')WRITE(6,116)
      IF(PRT.EQ.'Y')READ(5,11)IHD
      IF(IHD.NE.'Y')GO TO 21
      DO 22 J5=1,K
      WRITE(6,117)J5
      READ(5,118)CHAN1(J5),CHAN2(J5)
22    CONTINUE
21    CONTINUE
      WRITE(6,119)
      READ(5,10)IF2
      IF(IHD.EQ.'Y')WRITE(IF2,120)(CHAN1(KX),CHAN2(KX),KX=1,K)
      DO 23 J6=IA,IB
      KF=2*K
      WRITE(IF2,121)S(1,J6),(S(NX,J6),NX=2,KF,2)
 3    CONTINUE
```

```
      IF(IF2.NE.6)CALL CLOSE(IF2,1)
      IF(IF2.EQ.6)WRITE(6,122)IF2
   24 CONTINUE
      RETURN
   10 FORMAT()
   11 FORMAT(5A1)
  100 FORMAT(2(A6)
  101 FORMAT(5X,'ENTER NUMBER OF DATA FILES TO BE USED',
     *1X'(NO DEC)')
  102 FORMAT(5X,I5,' LIMIT, RE-ENTER NO. OF FILES.')
  103 FORMAT(5X,'SAME INPUT FORMAT FOR ALL FILES?')
  104 FORMAT(5X,'ENTER INPUT DATA FILE FORMAT DESIGNATION:',//,
     *5X,'1=IEDIT CODE, T, V, R (I5,3E20.8)',/,
     *5X,'2=T,V,IR,IEDIT CODE(E21.8,E20.8,2I20)',/,
     *5X,'  (OUTPUT FROM ENLI*TM EDIT OPTION)',/,
     *5X,'3="N" REAL VARIABLES & WRITE IN FORMAT',/,
     *5X,'4=UNFORMATTED INPUT, "N" REAL VARIABLES',)
  105 FORMAT(5X,'READ ONLY EDITED POINTS?')
  106 FORMAT(5X,'MAKE DATA FILE DESIGNATION AND BLOK',
     *1X,'FOR FILE',I5)
  107 FORMAT(I5,3E20.8)
  108 FORMAT(E21.8,E20.8,2I20)
  109 FORMAT(5X,'ENTER NUMBER OF VARIABLES IN INPUT DATA',
     *1X,'(NO DEC)')
  110 FORMAT(5X,'ENTER FORMAT STATEMENT INCLUDING BRACKETS.')
  111 FORMAT(5X,'ENTER INPUT DEVICE (UNIT)DESIGNATER (NO DEC)')
  112 FORMAT(5X,'DESIGNATE VARIABLE NUMBERS FOR X, Y (NO DEC)')
  113 FORMAT(5X,'FILE',I5,' CONTAINS',I5,' DATA POINTS')
  114 FORMAT(5X,'SORTED COUNT=',I5,/,5X,'FIRST TIME=',F10.3,',',
     *5X,'LAST TIME=',F10.3)
  115 FORMAT(5X,'ENTER FIRST, LAST POINT DESIGNATIONS (NO DEC)',/,
     *5X,'MINIMUM OF 1, MAXIMUM OF',I5)
  116 FORMAT(5X,'WANT TITLE DATA IN PRINTOUT?')
  117 FORMAT(5X,'ENTER CHANNEL NUMBER OR NUMBERS FOR FILE',I5)
  118 FORMAT(2A6)
```

```
119    FORMAT(5X,'ENTER "TMRD" OUTPUT FILE DESIGNATER (NO DEC)')
120    FORMAT('1',///,' PSEUDO TIME',5X,50(2A6))
121    FORMAT(1X,F12.3,50F12.0)
122    FORMAT(5X,'OUTPUT IN FILE',I5,' FORMAT(1X,F12.3,NF12.0)')
123    FORMAT(5X,'PRINT OUT SORTED DATA?')
       END
```

MDC-SDRF*F(1).PBV

```
      COMPILER(DIAG=3)
      SUBROUTINE PBV
      PARAMETER IPRA=12,IPRB=2500,IPRC=IPRA/2
      DOUBLE PRECISION SPH,CPH,SPS,CPS,ST,CT,X4,Y4,Z4
      DOUBLE PRECISION X3,Y3,Z3,X2,Y2,Z2,X1,Y1,Z1
      DOUBLE PRECISION AX4,AY4,AZ4
      COMMON S(IPRA,IPRB),K,KOUNT
      DIMENSION V(128)
C
C     COMPUTE BODY RELATED VELOCITY AND ACCELERATION DATA
C
      CALL ERTRAN(6,'@FREE 7. .')
      WRITE(6,1000)
      READ(5,10)IC
      IF(IC.LT.1.OR.IC.GT.1)WRITE(6,1001)
      IF(IC.LT.1.OR.IC.GT.1)READ(5,10)GR,GY,GP
      IF(IC.LT.1.OR.IC.GT.1)GO TO 20
      GR=1.
      GY=1.
      GP=1.
   20 CONTINUE
      WRITE(6,1002)
      READ(5,10)IVSF @ SCALE FACTOR "UNITS" DESIGNATER
      WRITE(6,1003)
      READ(5,10)GV @VELOCITY SCALE FACTOR
      WRITE(6,1004)
      READ(5,10)GT @FRAME TIME
      G=3.14159266/180.
      DO 200 I=2,KOUNT
      PH=S(2,I)*G/GR
      PS=S(4,I)*G/GY
      T=S(6,I)*G/GP
      TIME=S(1,I)
      SPH=DSIN(PH)
```

```
      CPH=DCOS(PH)
      SPS=DSIN(PS)
      CPS=DCOS(PS)
      ST=DSIN(T)
      CT=DCOS(T)
      X4=S(10,I)/GV
      Y4=S(12,I)/GV
      Z4=S(8,I)/GV
      I1=I-1
      GT1=(S(1,I)-S(1,I1))*GT
      AX4=((S(10,I)-S(10,I1)/GV)/GT1)
      AY4=((S(12,I)-S(12,I1)/GV)/GT1
      AZ4=((S(8,I)-S(8,I1))/GV)/GT1
      DO 100 J1=1,2
      GO TO(101,102),J1
102   CONTINUE
      X4=AX4
      Y4=AY4
      Z4=AZ4
101   CONTINUE
      X3=X4*CPH-Y4*SPH
      Y3=Y4*SPH+Y4*CPH
      Z3=Z4
      X2=X3
      Y2=Y3*CPS-Z3*SPS
      Z2=Y3*SPS+Z3*CPS
      X1=X2*CT+Z2*ST
      Y1=Y2
      Z1=-X2*ST+Z2*CT
      AMAG=SQRT(X1**2+Y1**2+Z1**2)
      Y11=-Y1
      X11=X1
      Z11=Z1
      CALL QUAD(X11,Y11,PHI)
      XX=SQRT(X11**2+Y11**2)
```

```
      CALL QUAD(XX,Z11,GAMMA)
      GO TO(103,104),J1
103   CONTINUE
      M=1
      V(1)=TIME
104   CONTINUE
      M1=M+1
      M2=M1+1
      M3=M2+1
      M4=M3+1
      M5=M4+1
      M6=M5+1
      M=M6
      V(M1)=X1
      V(M2)=Y1
      V(M3)=Z1
      V(M4)=AMAG
      V(M5)=PHI
      V(M6)=GAMMA
100   CONTINUE
      WRITE(7)(V(I2),I2=1,13)
200   CONTINUE
      CALL CLOSE(7,1)
C
C     WRITE OUT HEADINGS, DATA, AND PLOT FILES
C
      WRITE(6,1005)
      READ(5,10)IPB
      IF(IPB.EQ.0.OR.IPB.LT.0)GO TO 121
      IF(IPB.GT.KOUNT)IPB=KOUNT
      IF(IPB.GT.0)WRITE(6,1006)
      IF(IPB.GT.0)READ(5,10)IF2
      IF(IPB.GT.0)WRITE(IF2,1008)
      IF(IVSF.EQ.1)WRITE(IF2,1009)GV,GT
```

```
      IF(IVSF.EQ.2)WRITE(IF2,1010)GV,GT
      KT=0
119   CONTINUE
      READ(7,END=120)(V(I3),I3=1,13)
      KT=KT+1
      IF(KT.GT.IPB)GO TO 120
      WRITE(IF2,1007)(V(I4),I4=1,13)
      GO TO 119
120   CONTINUE
      IF(IF2.EQ.6)GO TO 121
      CALL CLOSE(IF2,1)
      WRITE(6,1011)IF2
121   CONTINUE
      WRITE(6,1008)
      IF(IVSF.EQ.1)WRITE(6,1009)GV,GT
      IF(IVSF.EQ.2)WRITE(6,1010)GV,GT
      WRITE(6,1012)
118   CONTINUE
      RETURN
10    FORMAT(
1000  FORMAT(5X,'INPUT GIMBAL DATA IN COUNTS OR',
     *' DEGREES? COUNTS=1, DEGREES=0')
1001  FORMAT(5X,'ENTER INNER, MIDDLE, OUTER GIMBAL SCALE',
     *' SCALE FACTOR, CTS/DEG., (INCL. DEC.)')
1002  FORMAT(5X,'VELOCITY SCALE FACTOR IN CTS/M/SEC=1, CTS/FT/SEC=2')
1003  FORMAT(5X,'ENTER VELOCITY SCALE FACTOR, INCL. DEC.')
1004  FORMAT(5X,'ENTER FRAME TIME')
1005  FORMAT(5X,'PBU PRINT OUTPUT? ENTER NO. OF PTS. (NO DEC.)')
1006  FORMAT(5X,'ENTER PBV OUTPUT FILE DESIGNATER (NO DEC.)')
1007  FORMAT(1X,F8.3,2(4G10.4,2F7.1))
1008  FORMAT('1',5X,'OUTPUT VARIABLE (COLUMN) IDENTIFICATION:',/,
     *5X,'1-PSEUDO TIME',/,
     *5X,'2,3,4-VEHICLE XDOT, YDDOT, ZDOT',/,
     *5X,'5-TOTAL VELOCITY MAGNITUDE',/,
```

```
     *5X,'6-ANGLE OF TOTAL VELOCITY VECTOR RELATIVE ',/,
     *10X,'TO (+ LEFT OF) VEHICLE VERTICAL LONGITUDINAL PLANE (PHI)',/,
     *5X,'7-ANGLE OF TOTAL VELOCITY VECTOR RELATIVE ',/,
     *10X,'TO VEHICLE HORIZONTAL PLANE (GAMMA)',/,
     *5X,'8-14 SAME AS 2-7 EXCEPT FOR ACCELERATION',/)
1009  FORMAT(5X,'VELOCITY SCALE FACTOR IS',F10.3,' CTS/METER/SEC ',
     *'FRAME TIME IS',F10.6,' SEC.',/)
1010  FORMAT(5X,'VELOCITY SCALE FACTOR IS',F10.3,' CTS/FT/SEC.  ',
     *'FRAME TIME IS',F10.6,' SEC.',/)
1011  FORMAT(5X,'OUTPUT IN FILE',I7)
1012  FORMAT(5X,'UNFORMATTED PLOT OUTPUT ON UNIT 7')
      END
```

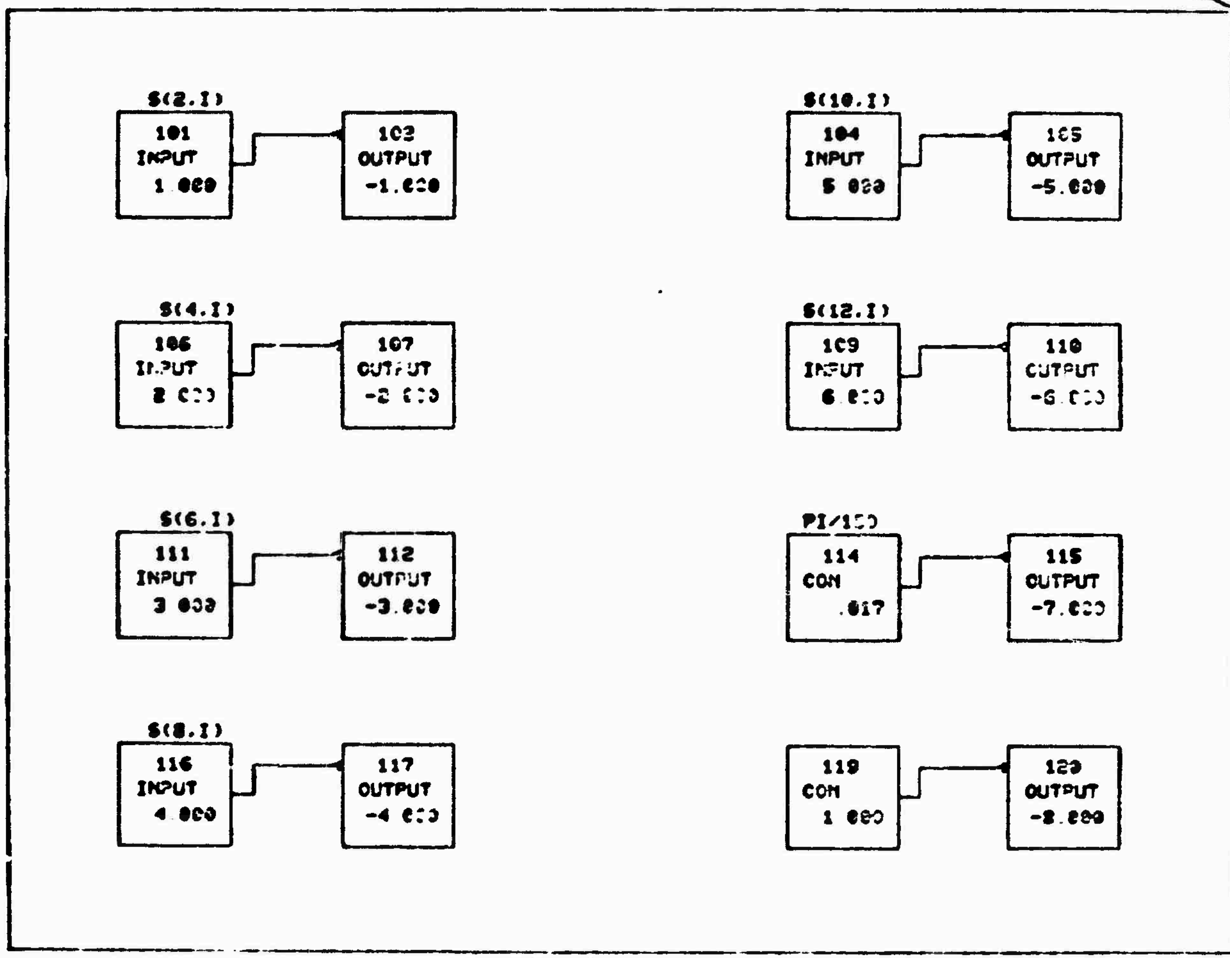
MDC-SDBG*G(1).QUAD

```
      SUBROUTINE QUAD(X,Y,C)
      PI=3.141592653589793 2/180.
      V=SQRT(X**2+Y**2)
      A=X/V
      B=Y/V
      IF(A)30,31,31
31    C=ASIN(A)/PI
      GO TO 34
30    IF(B)32,33,33
33    C=180.-ASIN(B)/PI
      GO TO 34
32    C=-180.-ASIN(B)/PI
34    RETURN
      END
```

SECTION B.2.

ALGORITHM NO. 1 IPAA DIAGRAMS

PAGE 1 [illegible]
[illegible] 8 1 VELOCITY [illegible]



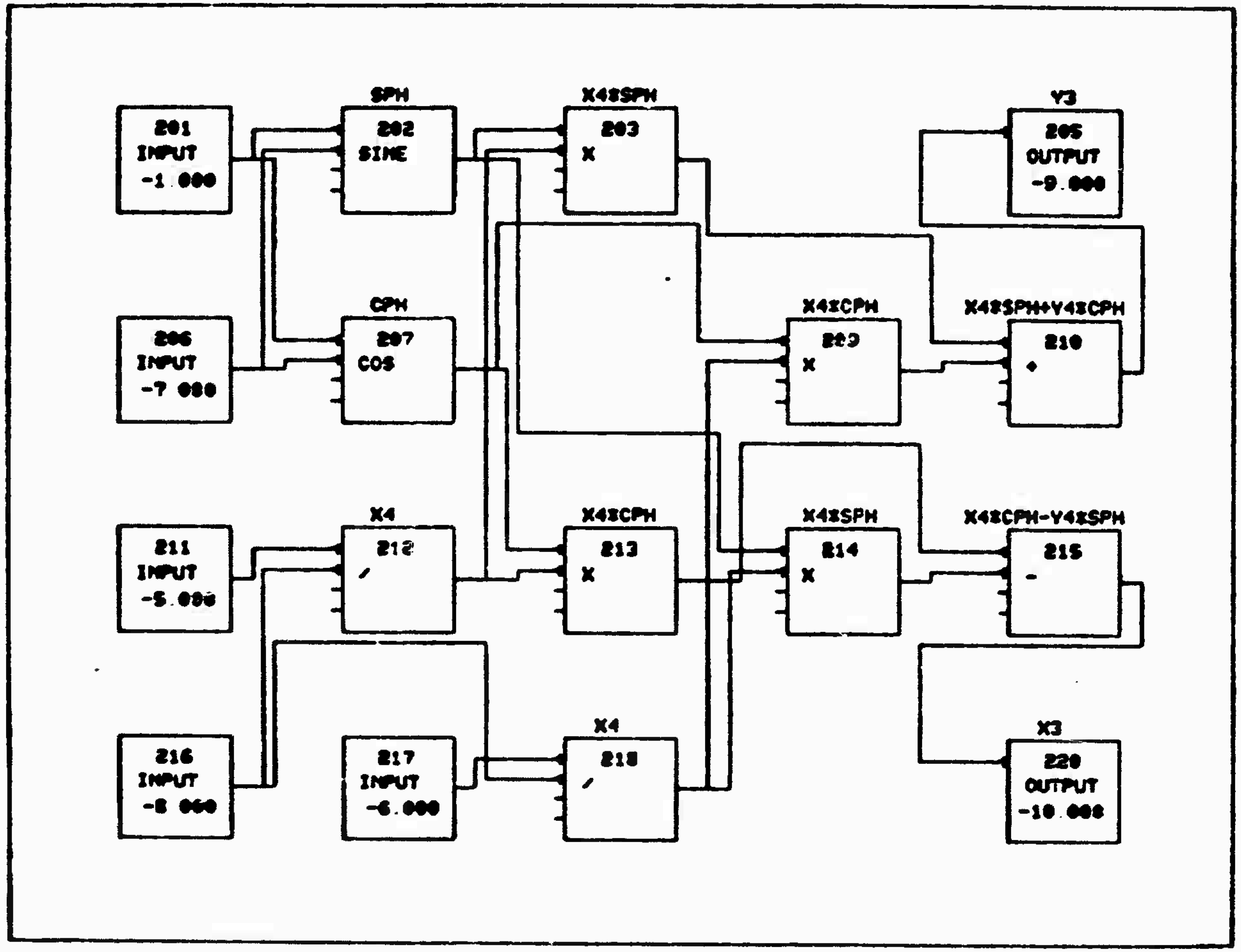
FUNCTIONS AVAILABLE:

INPUT	OUTPUT
TIME	DELAY
CON	GAIN
INTG1	INTG2
+	-
X	/
SINE	COS
ATAN	EXP
ABS	MOD
BHOLD	FCIN1
DISP	CAL
SIMP	ASIN
ACOS	TAN
SQRT	POWER
LOGE	LOG10

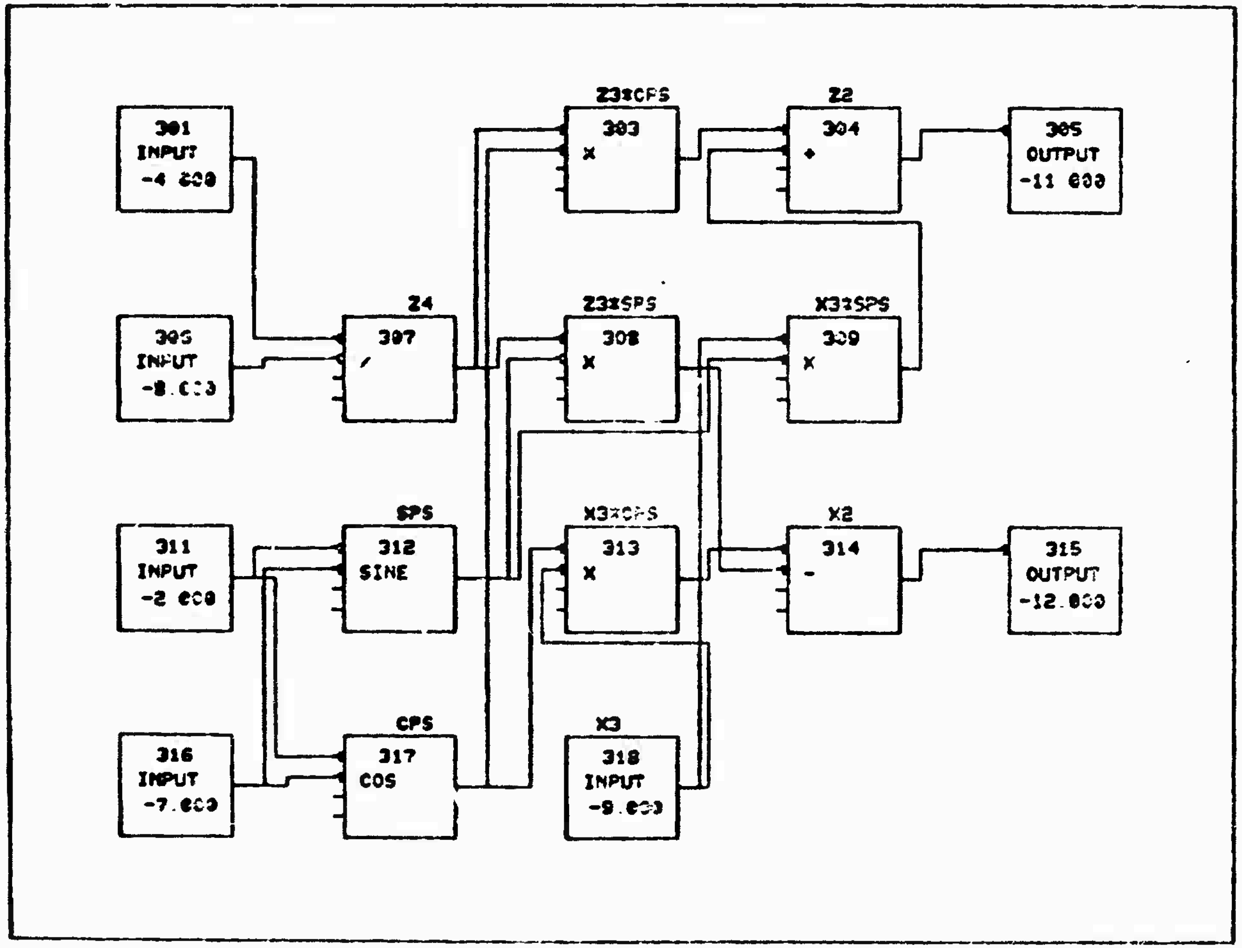
KEY OPTIONS:

- *BLOCK DEF
- A=[illegible] STATE
- [illegible]
- C=[illegible]ECT
- D=[illegible]
- E=[illegible] ADD[illegible] N[illegible]T
- F=FL[illegible] DEF
- K=CHECK DIA
- L=LAST DIAG
- M=[illegible] PAR
- P=[illegible]AM PRT
- R=RECALL
- S=SAVE
- T=TEXT INP
- Z=ZOOM
- 1-9=PAGE
- X=EXECUTE DIAG
- CNTL-C=EXIT

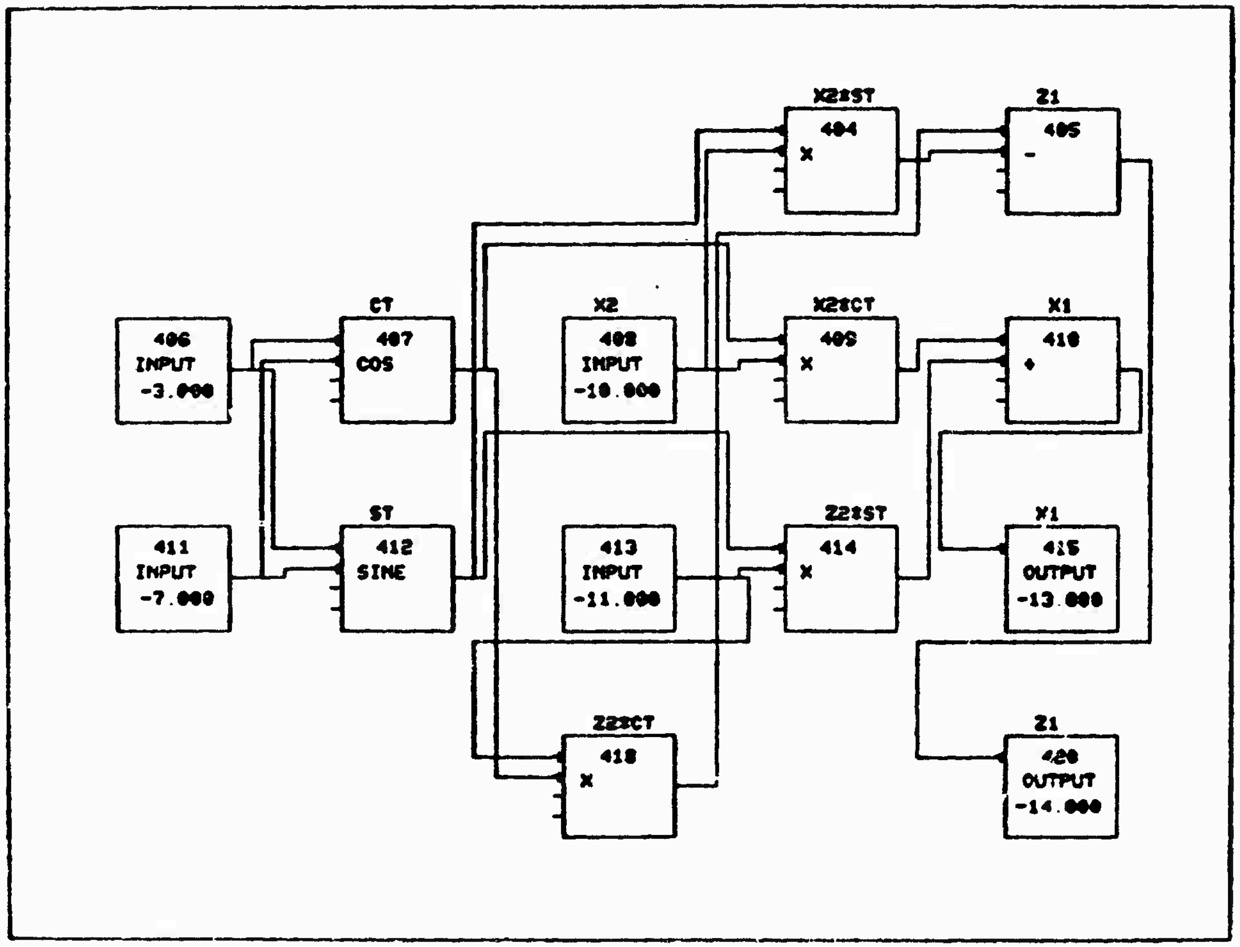
PAGE 2 IPAA
ALGO # 1 VELOCITY SECTION



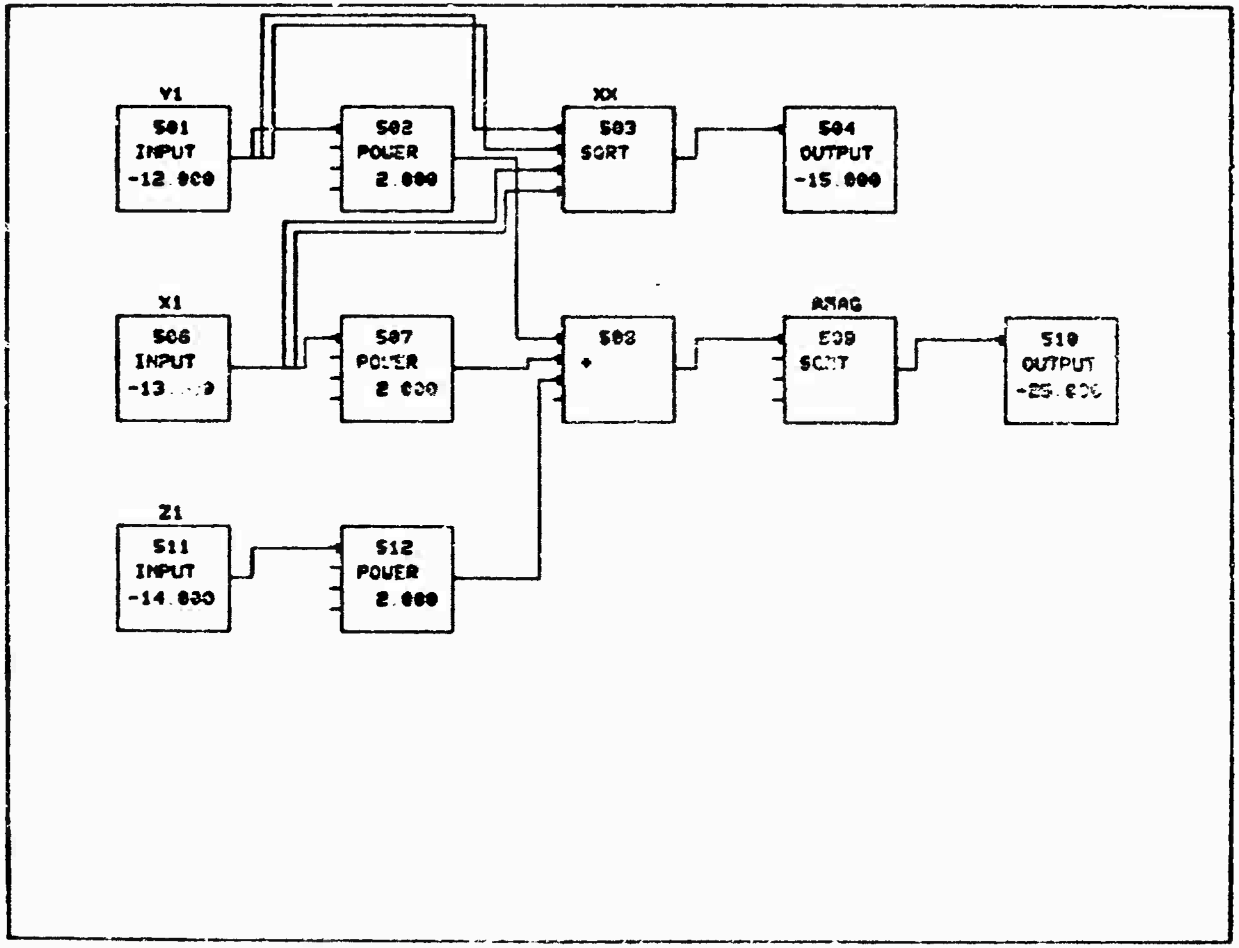
PAGE 3 IPAD
AICO # 1 VELOCITY SECTION



PAGE 4 IPAA
ALGO $ 1 VELOCITY SECTION

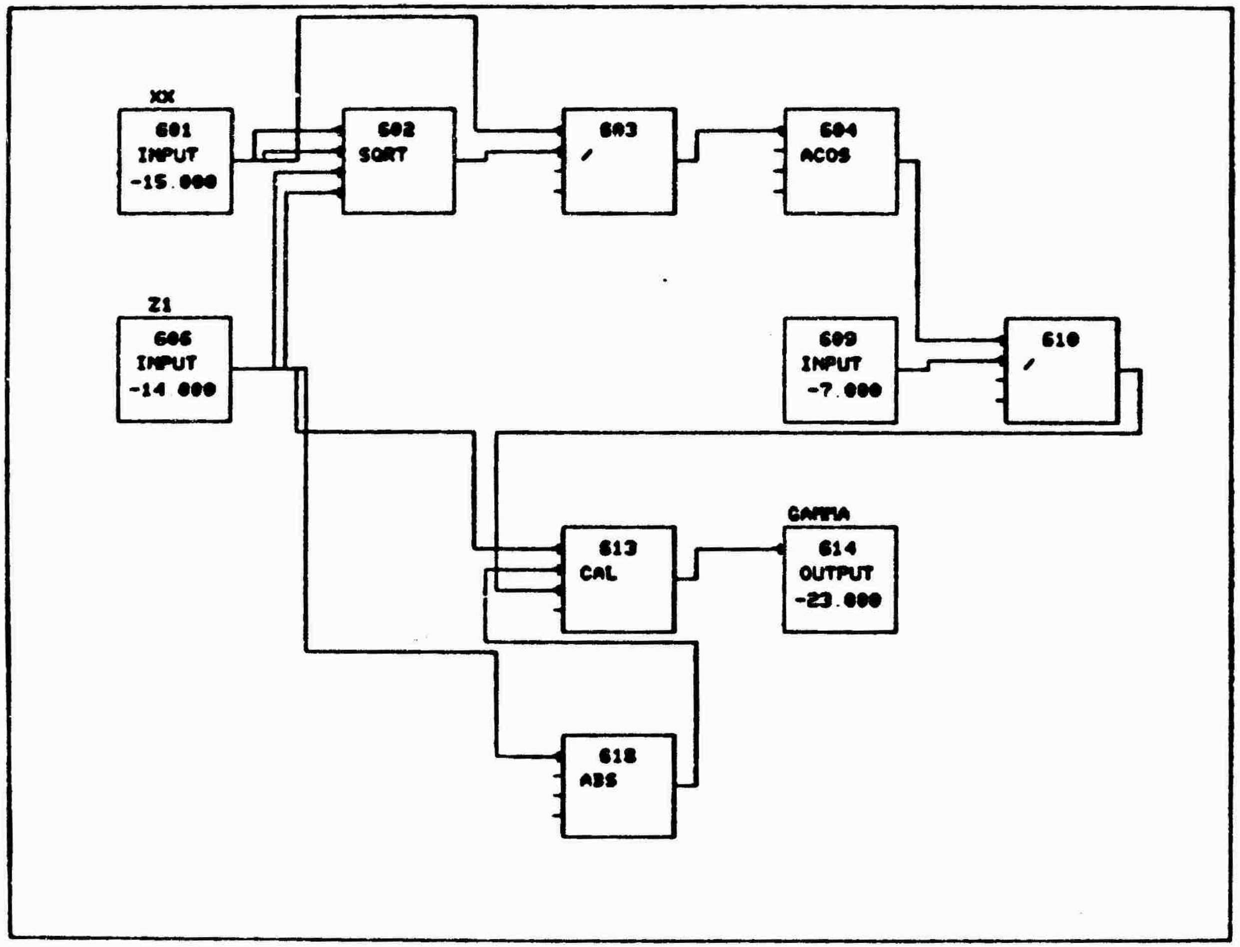


PAGE 5 IPRA
ALGO. # 1 VELOCITY SECTION



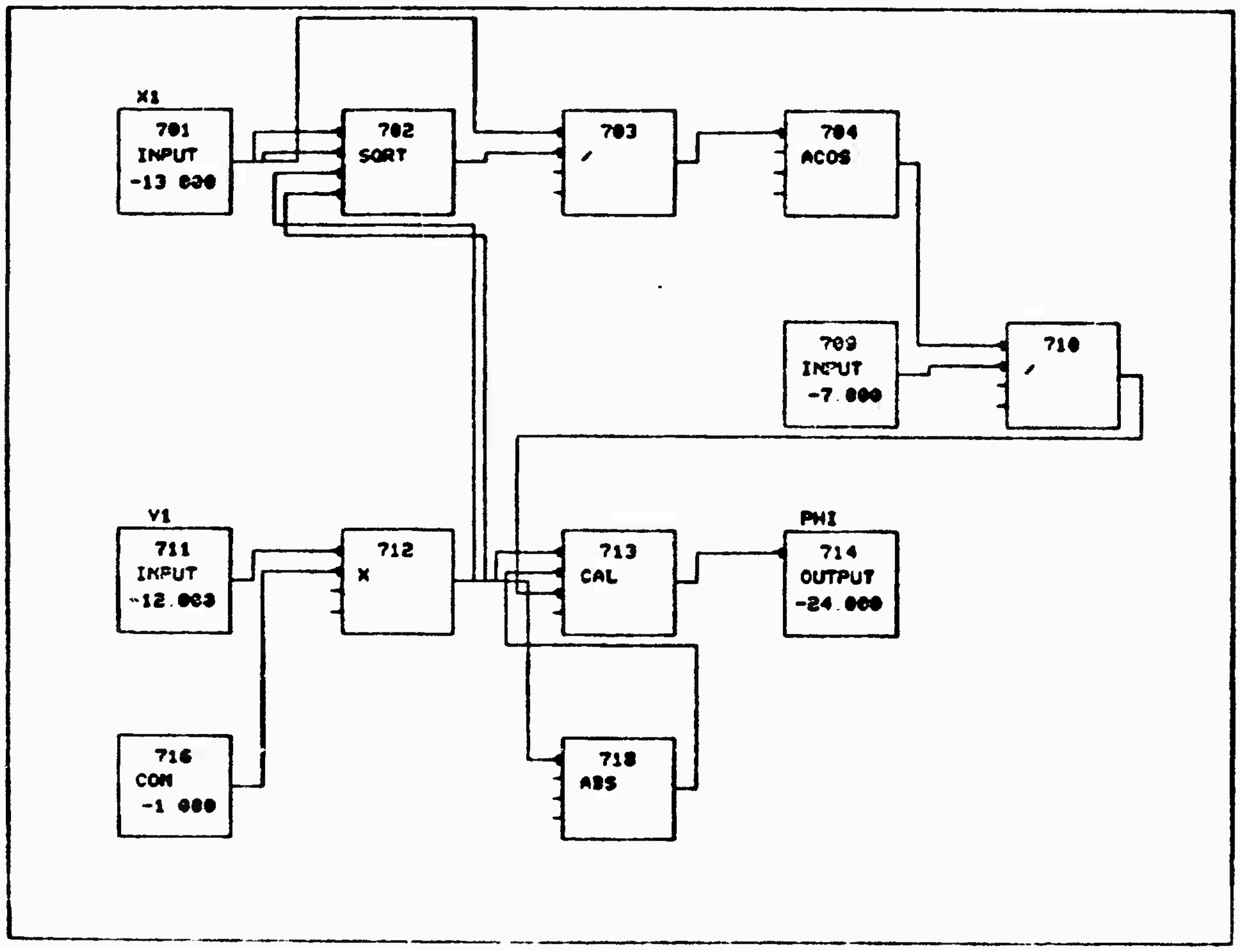
BLOCK 613
P(1)= 4.000
P(2)= 3.000
P(3)= 0.000

PAGE 6 IPAA
ALGO. 8 1 VELOCITY SECTION

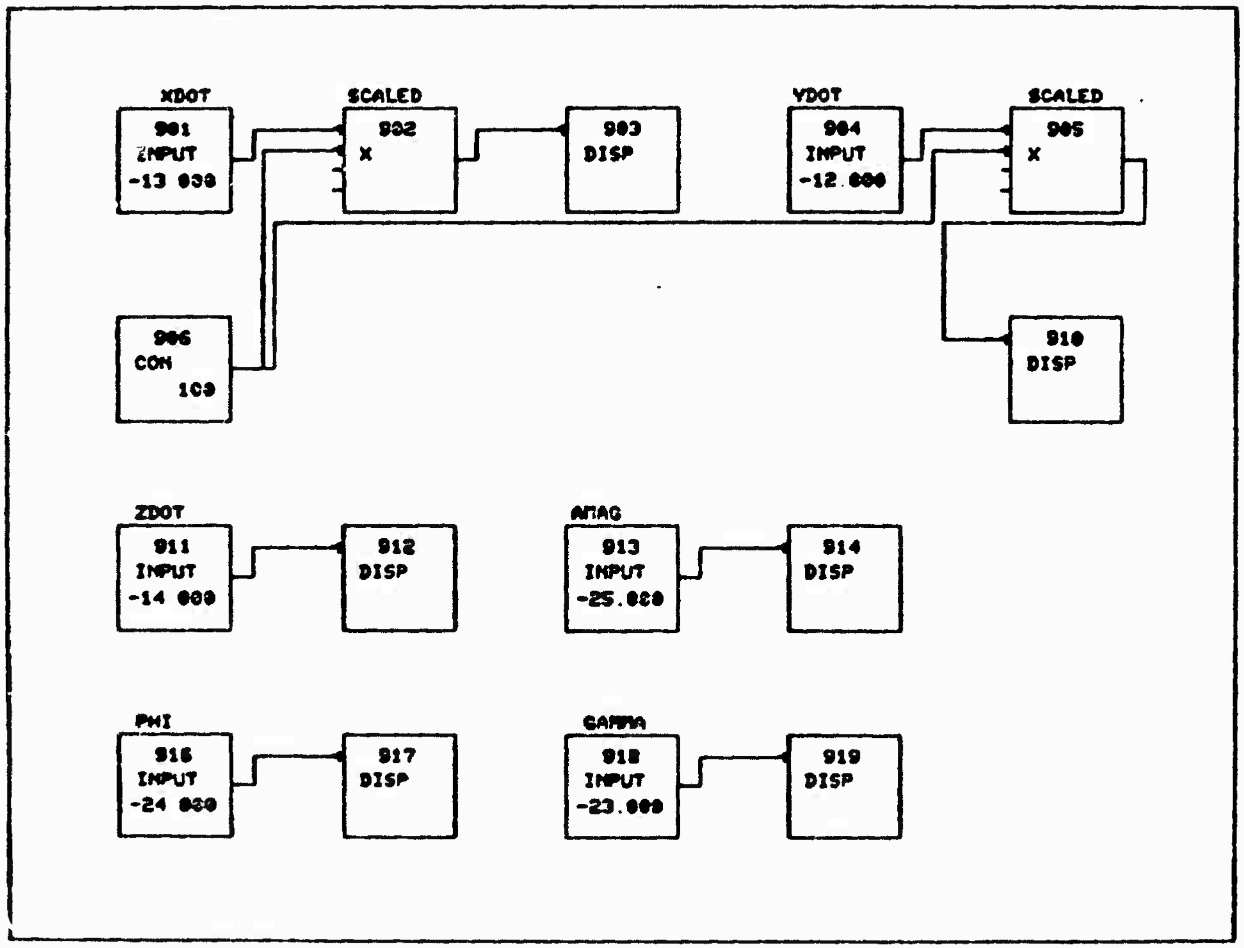


CLOCK 713
P(1)= 4.000
P(2)= 3.000
P(3)= 0.000

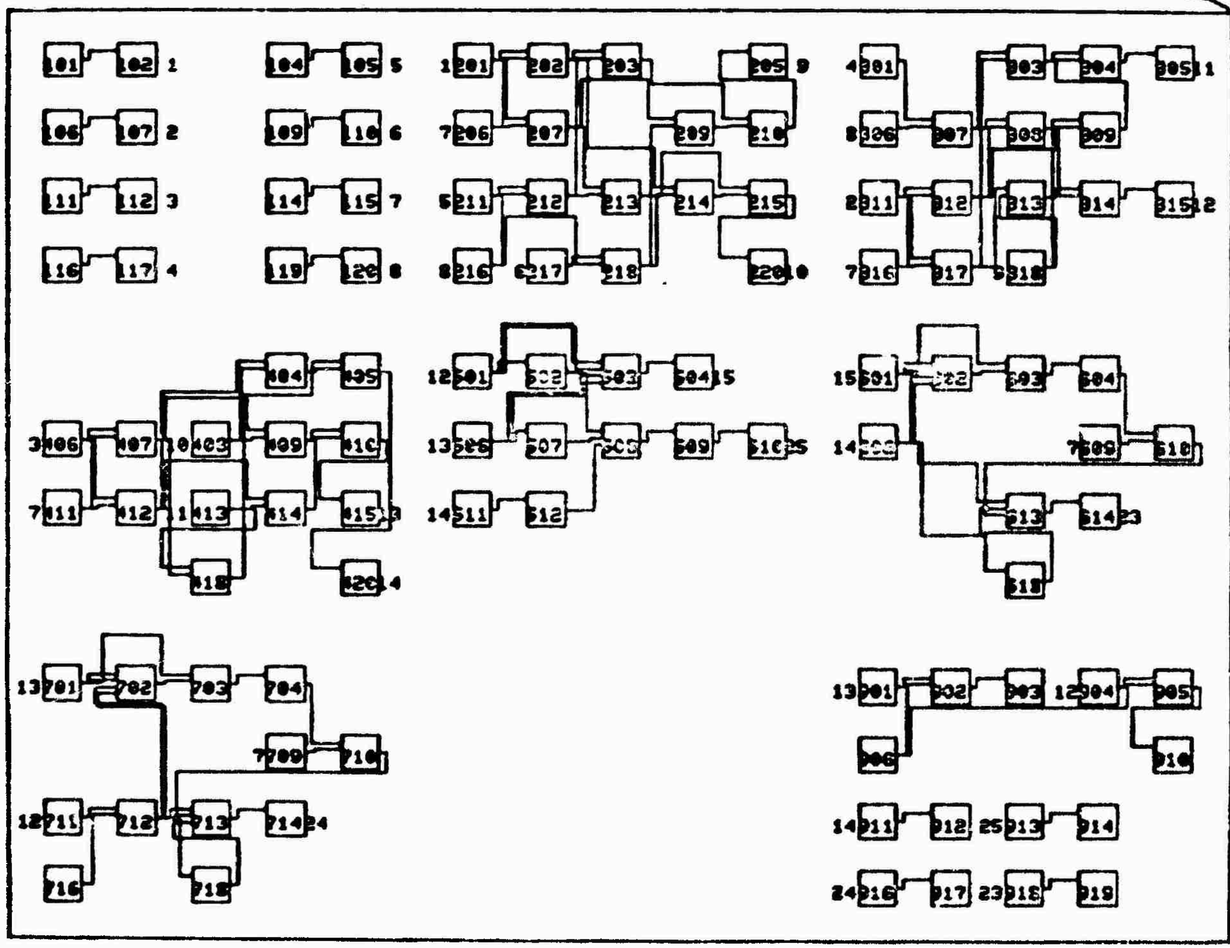
PAGE 7 IPAA
ALGO. # 1 VELOCITY SECTION



PAGE 9 IPRA
ALCO 8 1 VELOCITY SECTION



IPAA ZOOM
ALGO. 8 1 VELOCITY SECTION



FUNCTIONS AVAILABLE:

INPUT	OUTPUT
TIME	DELAY
CON	GAIN
INTG1	INTG2
+	-
X	/
SINE	COS
ATAN	EXP
ABS	MOD
SHOLD	FGEN1
DISP	CAL
SIMP	ASIN
ACOS	TAN
SQRT	POWER
LOGE	LOG10

KEY OPTIONS:

-BLOCK DEF
A-ANNOTATE
ABCD-PINS
C-CONNECT
D-REDRAW
E-ERASE- ABCDNT
N-NET
F-FUNC. DEF
K-CHECK DIA
L-LAST DIAG
M-MOD PAR
P-PARAM PRT
R-RECALL
S-SAVE
T-TEXT INP
Z-ZOOM
1-9-PAGE
X-EXECUTE DIAG
CNTL-C-EXIT

OUTPUT ALGO # 1 (VELOCITY)

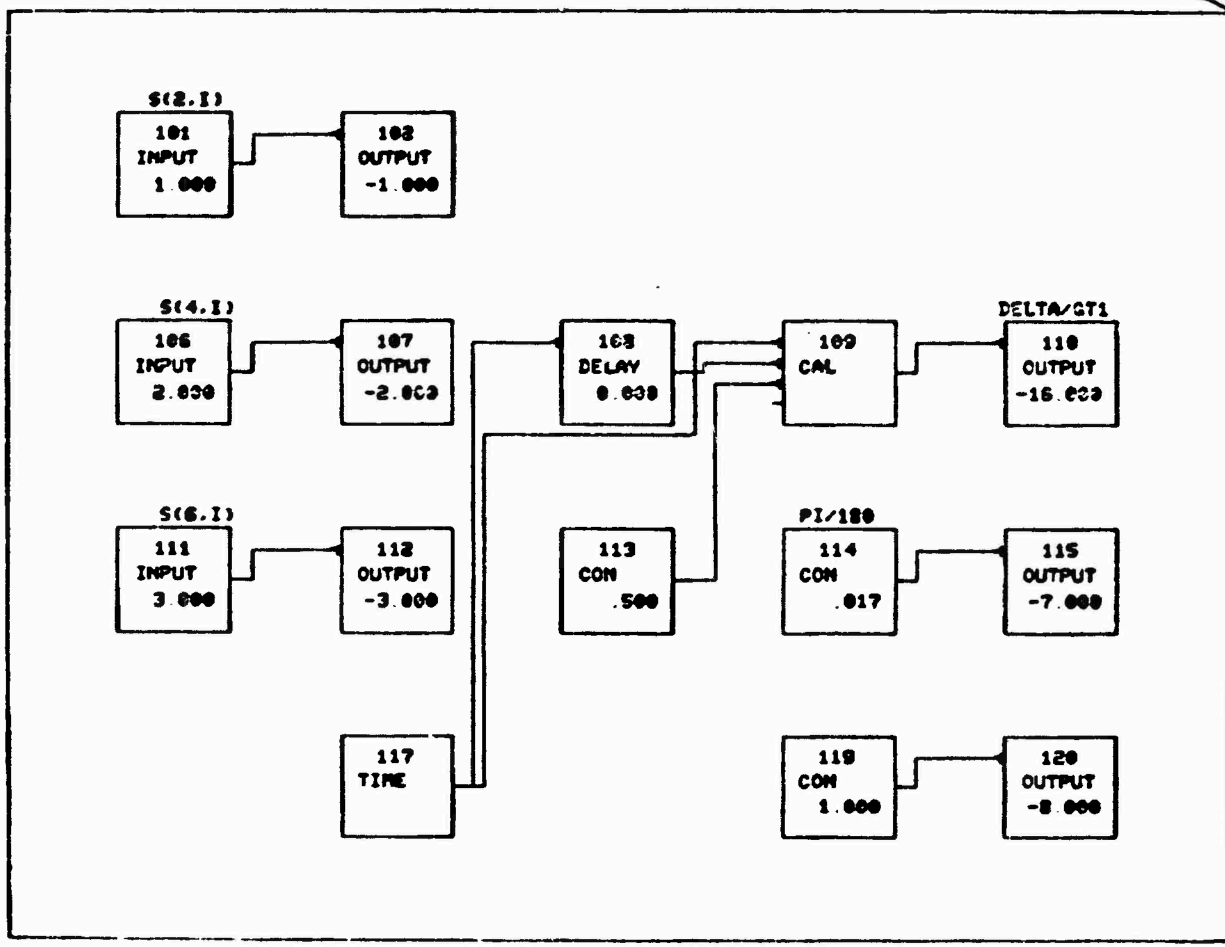
** IPAA PROCESSOR INFORMATION **

```
1-  PROCESSING DESCRIPTION.

2-  DIAGRAM NAME                    ALGOV.
3-  START TIME                         1.000
4-  STOP TIME                         41.000
5-  TIME INCREMENT                     1.000
6-  INPUT INTERPOLATION DEGREE      1
7-  DISPLAY RATIO                    1
8-  TIME OPTION                     TIME INPUT PROVIDED
-------
INPUTS.
9-  INPUT FORMAT                    FORMATTED DATA
                                    FILE NAME                          START TIME
10- INPUT FILE NO. 1                GIM3AL2.                             0.000
11- NUMBER OF FIELDS                 7
12- INPUT                            T 1 2 3 4 5 6
    REFERENCE                        1 2 3 4 5 6 7
13- FIELD                            1 2 3 4 5 6 7
    WIDTH                           15151515151515
OUTPUT
------
14- OUTPUT FORMAT                   DEFAULT FORMAT(POT)
                                    FILE NAME
15- OUTPUT FILE NO. 1
16- OUTPUT FILE NO. 2
17- OUTPUT FILE NO  3
18- OUTPUT FILE NO. 4
20- SAVE PIF IN FILE
21- RESTORE PIF FROM FILE
22- RETURN TO DIAGRAM
30- EXECUTE
```

BLOCK 109
P(1)= 2.000
P(2)= 3.000
P(3)= 0.000

PAGE 1 IPAA
ALGO. # 1 ACCELERATION SEC.



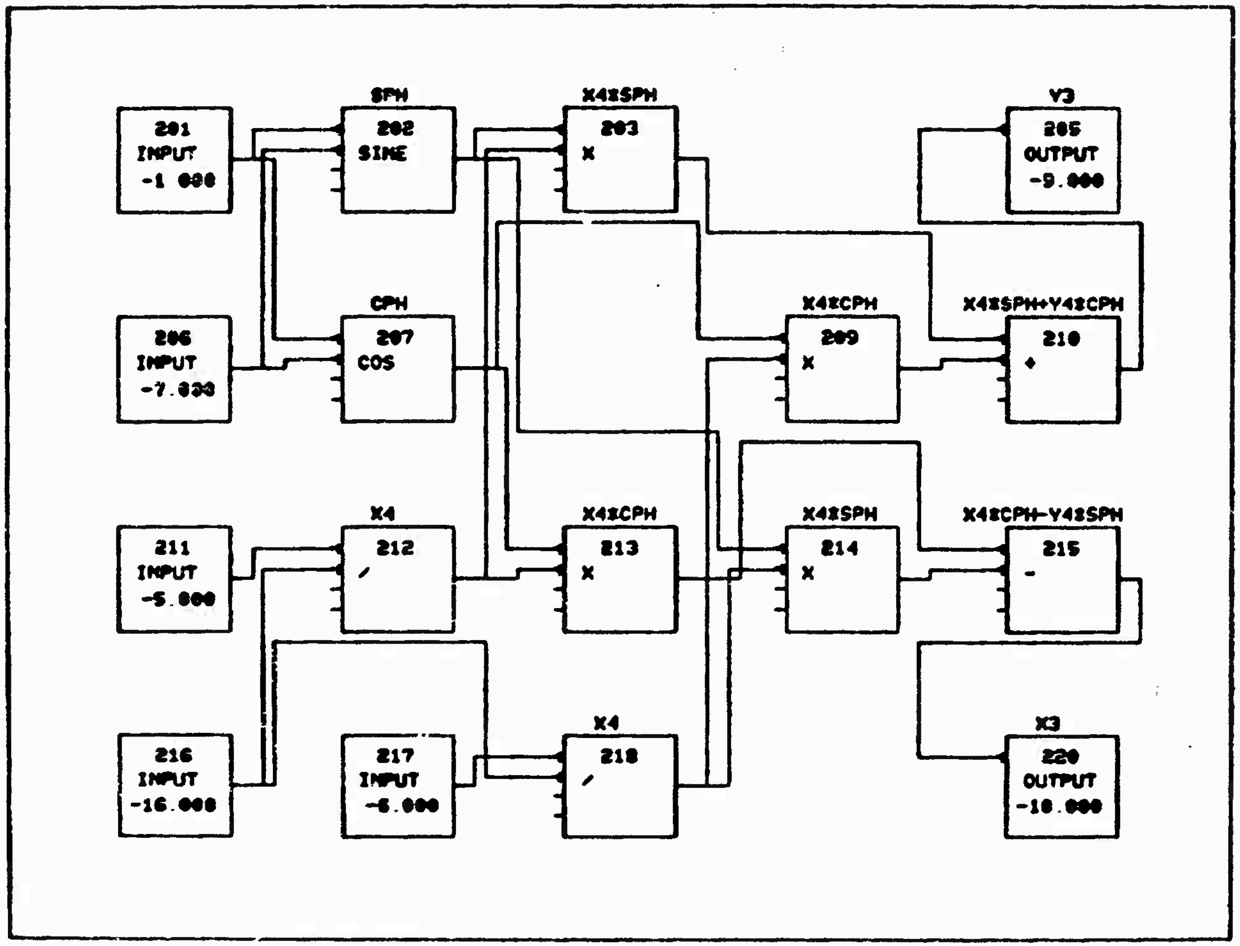
FUNCTIONS AVAILABLE:

INPUT	OUTPUT
TIME	DELAY
CON	GAIN
INTG1	INTG2
+	-
X	/
SINE	COS
ATAN	EXP
ABS	MOD
CHOLD	FGEN1
DISP	CAL
SIMP	ASIN
ACOS	TAN
SQRT	POWER
LOGE	LOG10

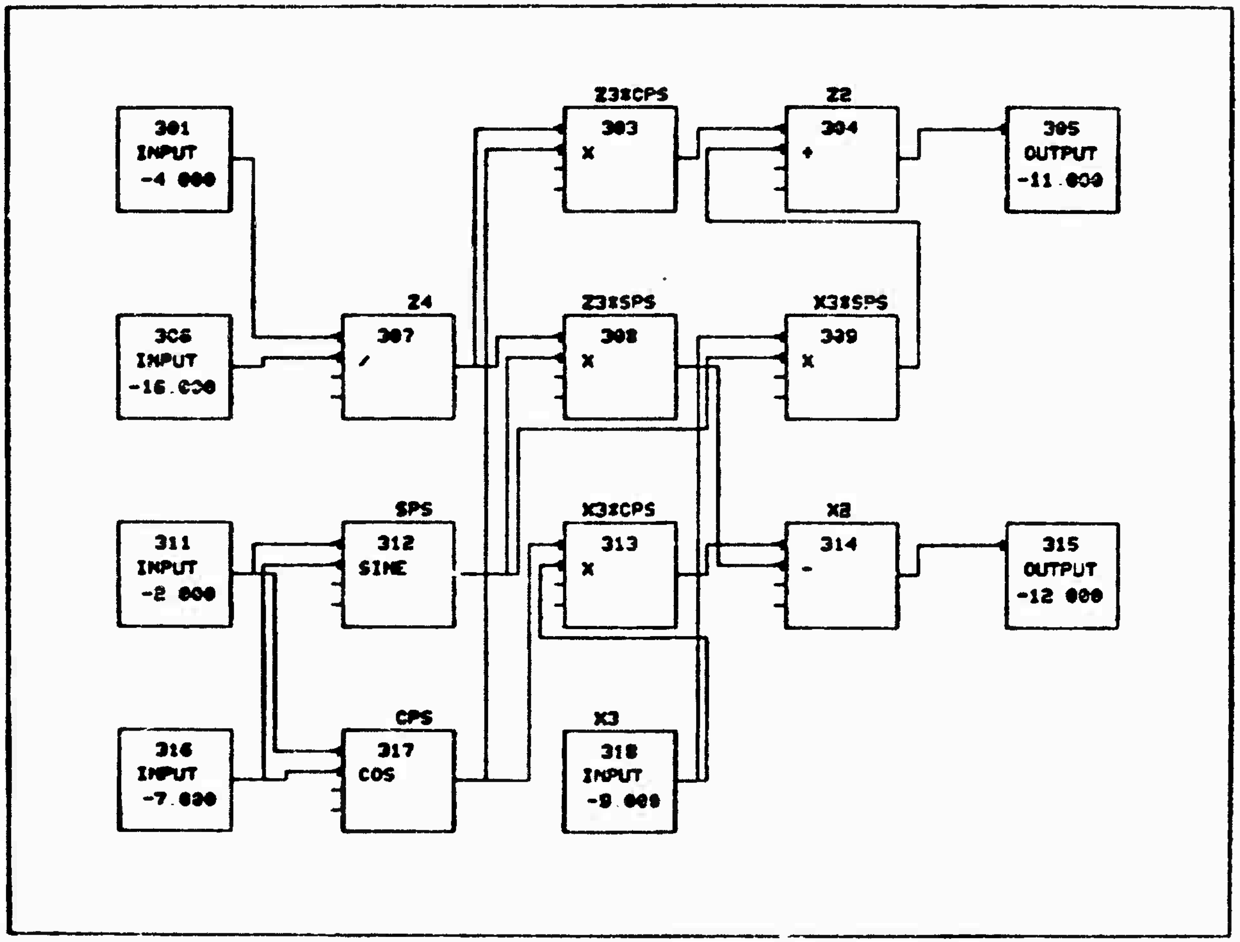
KEY OPTIONS:

=BLOCK DEF
A=ANNOTATE
ADCD=PINS
C=CONNECT
D=REDRAW
E=ERASE= ADCDNT
N=NET
F=FUNC. DEF
K=CHECK DIA
L=LAST DIAG
M=MOD PAR
P=PARAM PRT
R=RECALL
S=SAVE
T=TEXT INP
Z=ZOOM
1-9=PAGE
X=EXECUTE DIAG
CNTL-C=EXIT

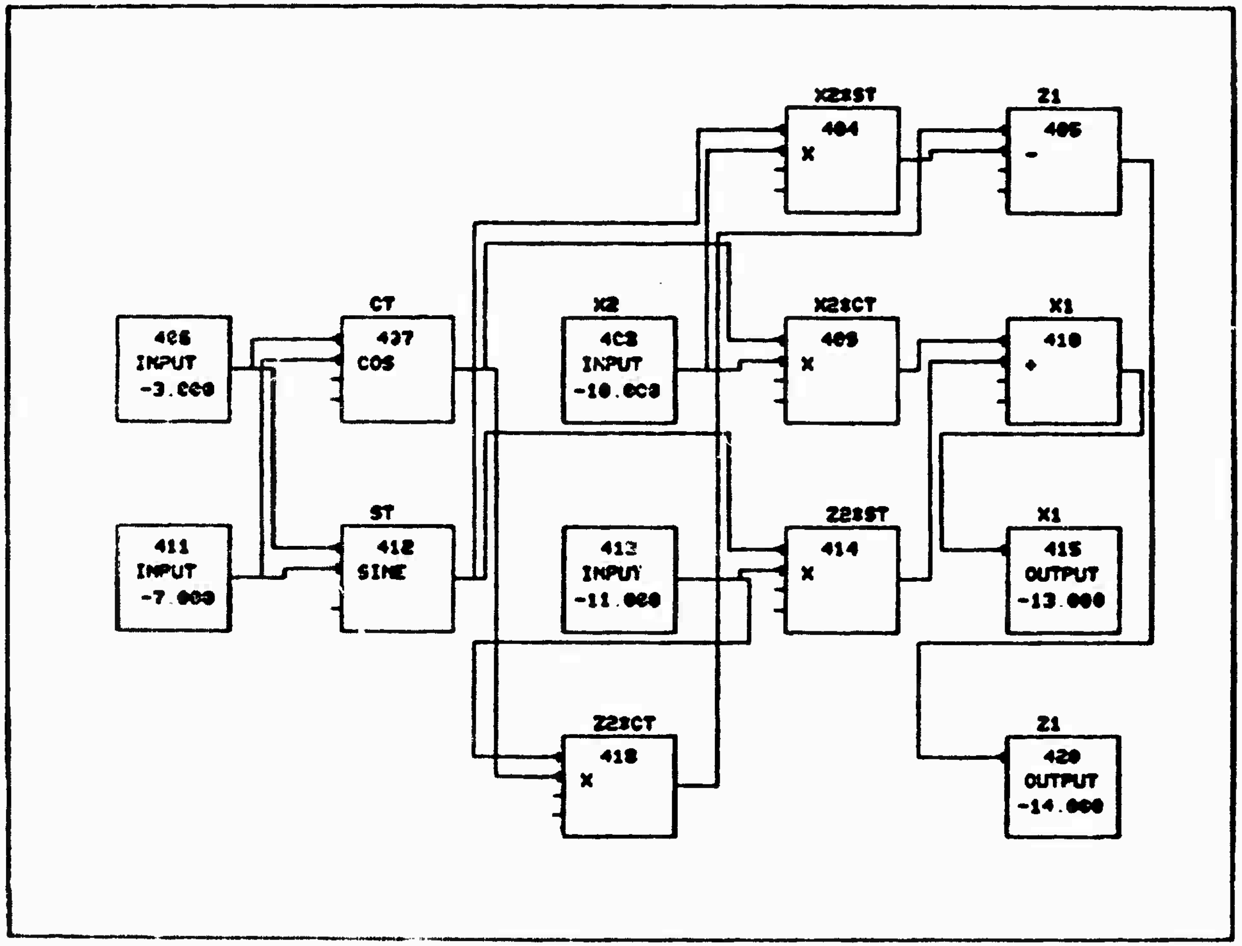
PAGE 2 IPAA
ALGO. # 1 ACCELERATION SEC.



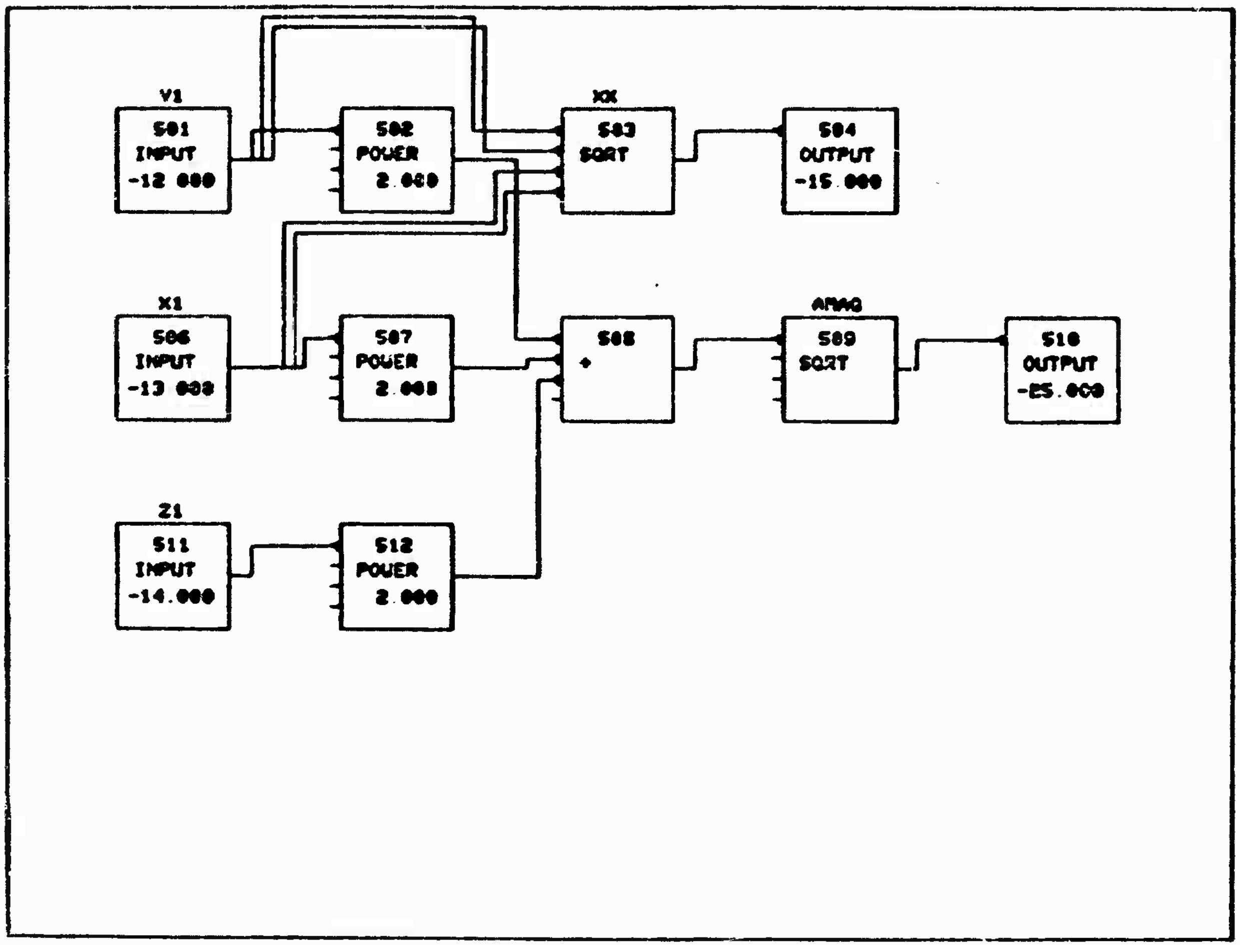
PAGE 3 IPAA
ALGO. # 1 ACCELERATION SEC.



PAGE 4 IPAA
ALGO 8 1 ACCELERATION SEC.

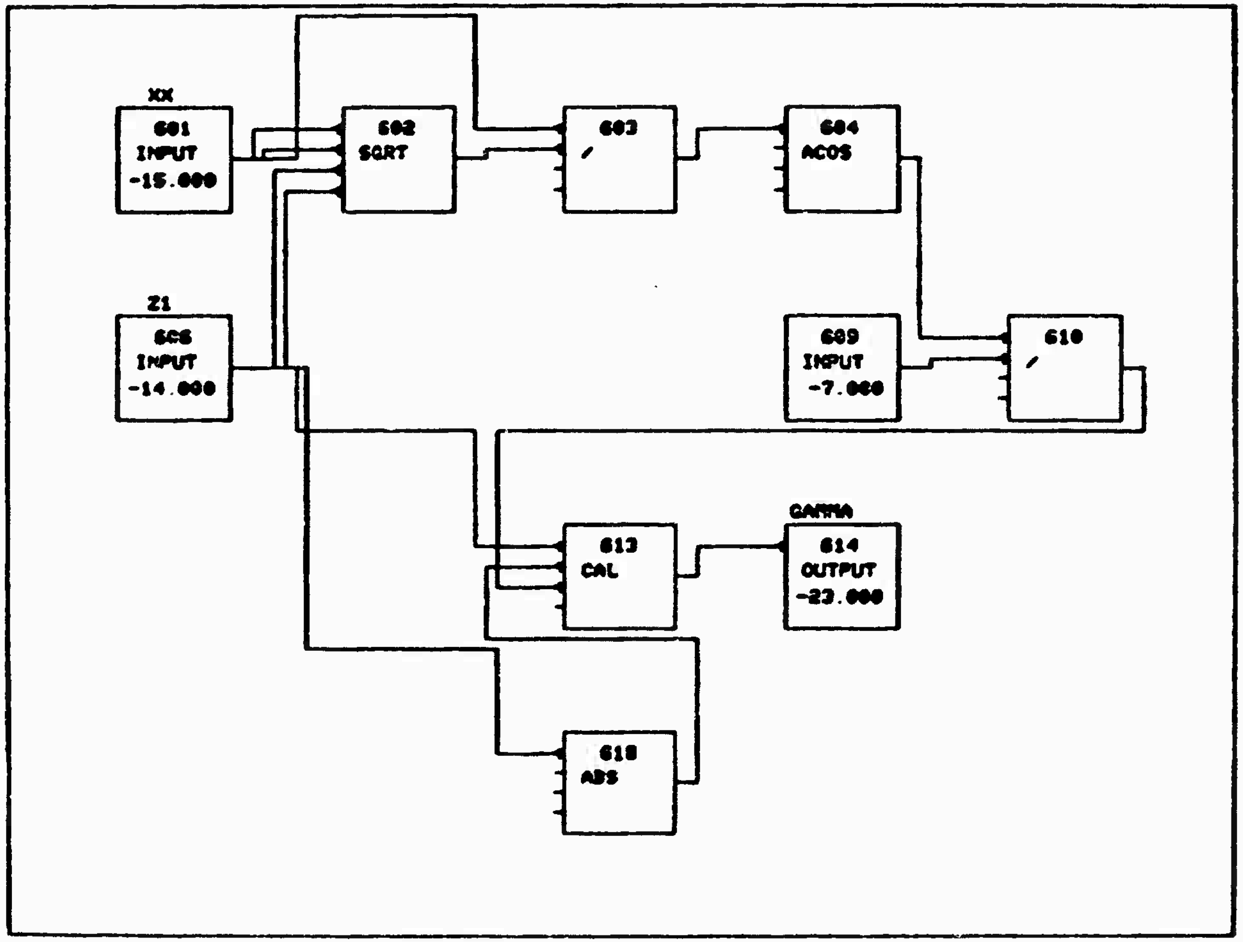


PAGE 5 IPAA
ALGO. 8 1 ACCELERATION SEC.



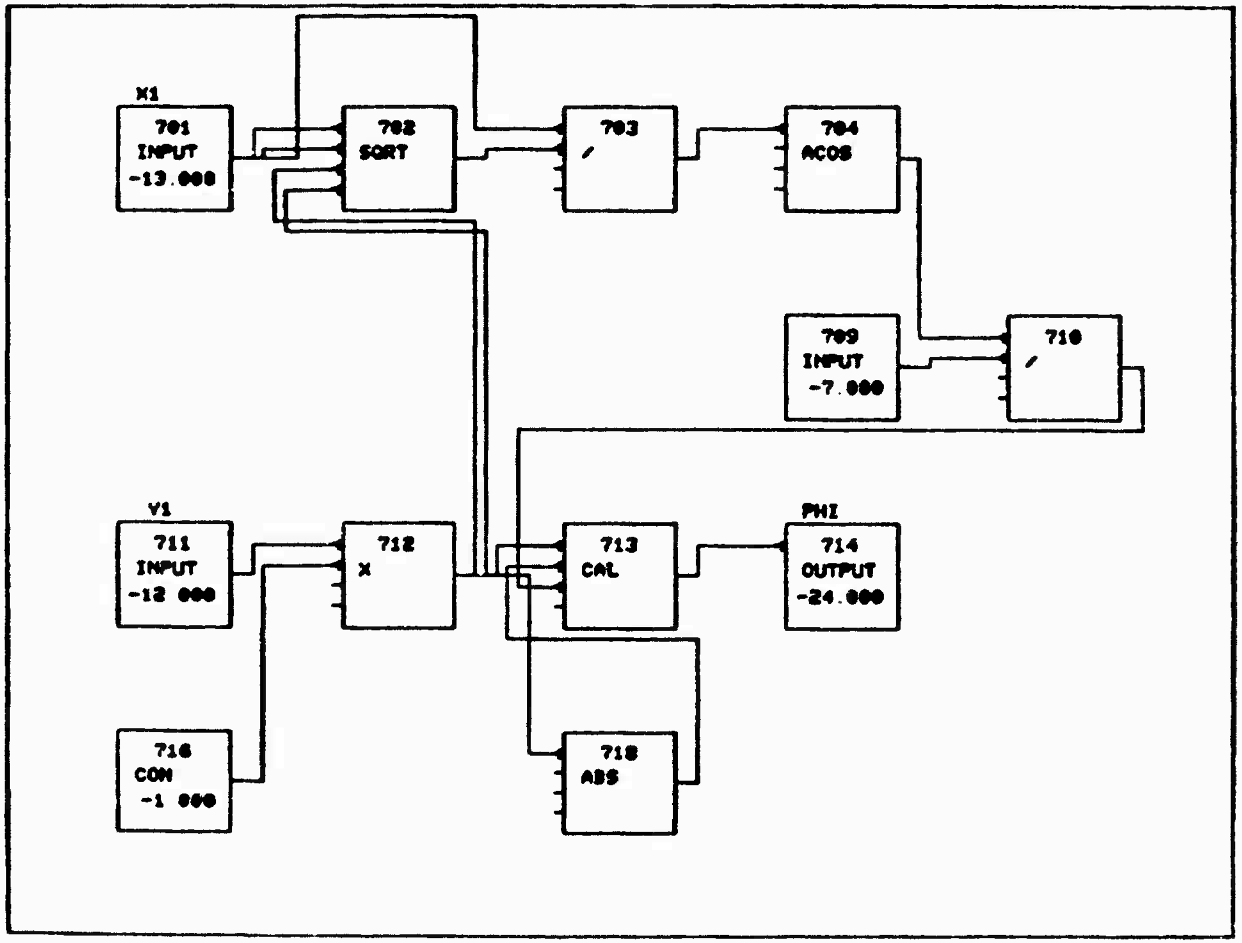
BLOCK 613
P(1)= 4.000
P(2)= 3.000
P(3)= 0.000

PAGE 6 IPAA
ALGO 8 1 ACCELERATION SEC.



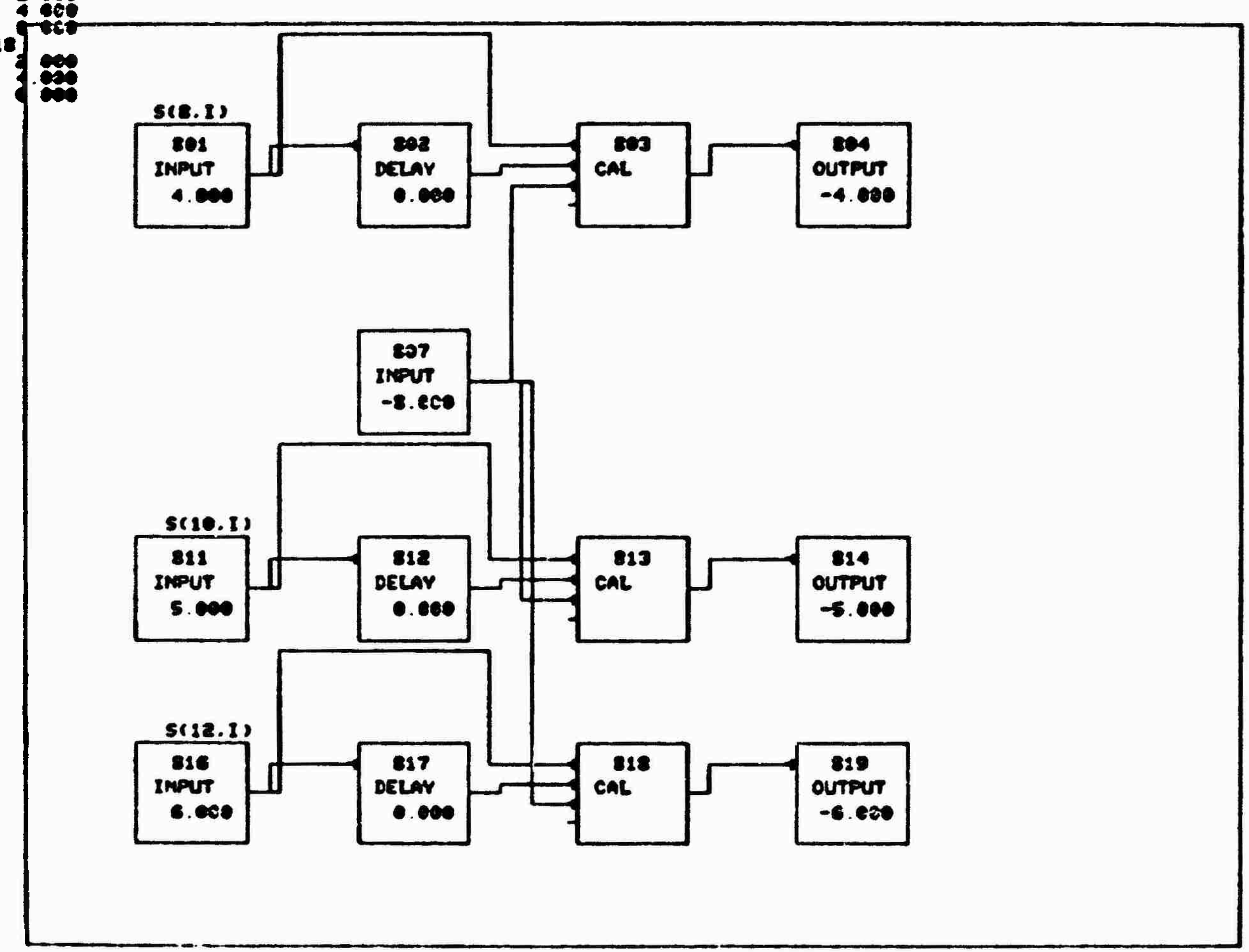
BLOCK 712
P(1)= 4.000
P(2)= 3.000
P(3)= 0.000

PAGE 7 IPAA
ALGO. # 1 ACCELERATION SEC.

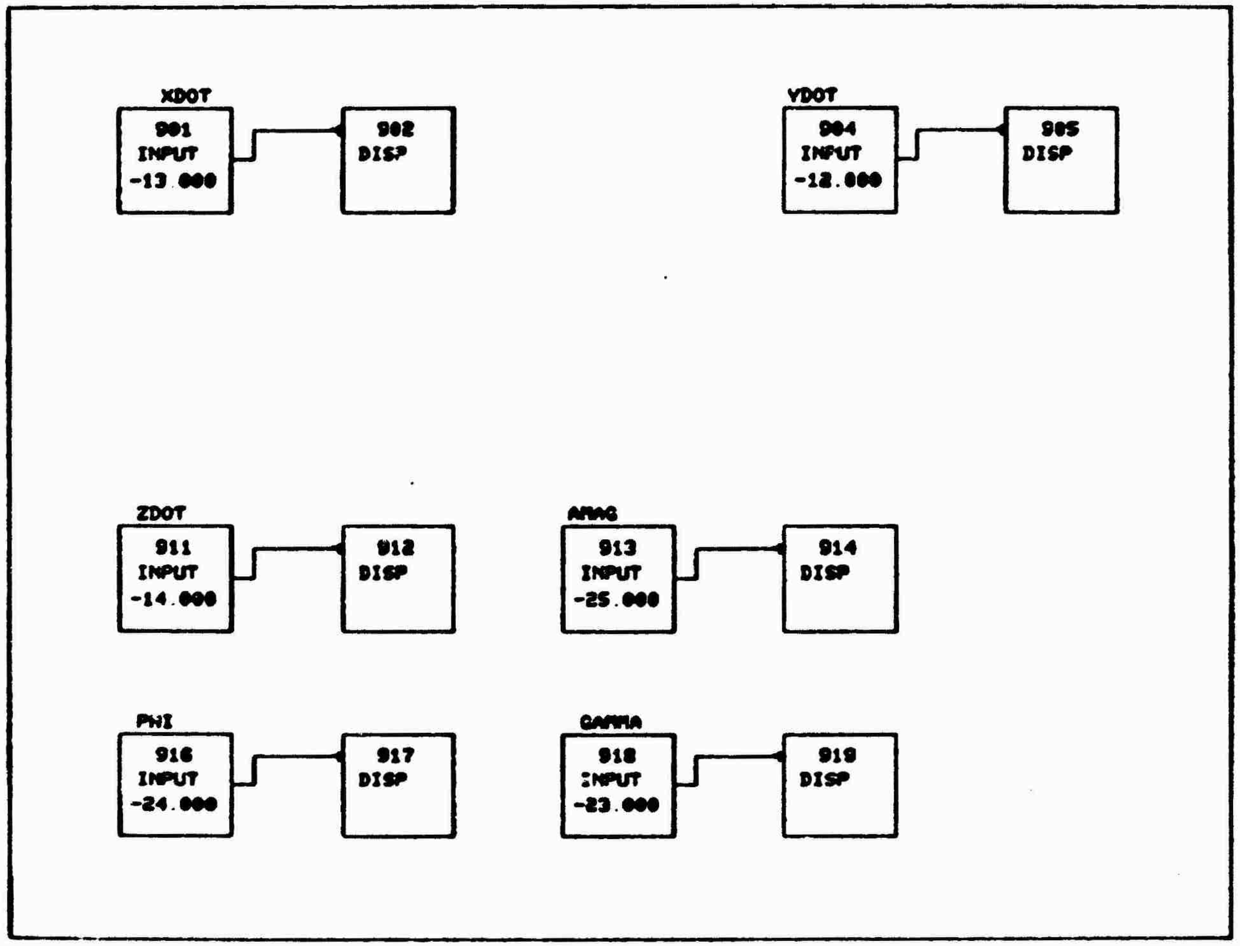


```
BLOCK 803
P( 1)=    2.000
P( 2)=    4.000
P( 3)=    0.000
BLOCK 813
P( 1)=    2.000
P( 2)=    4.000
P( 3)=    0.000
BLOCK 818
P( 1)=    2.000
P( 2)=    4.000
P( 3)=    0.000
```

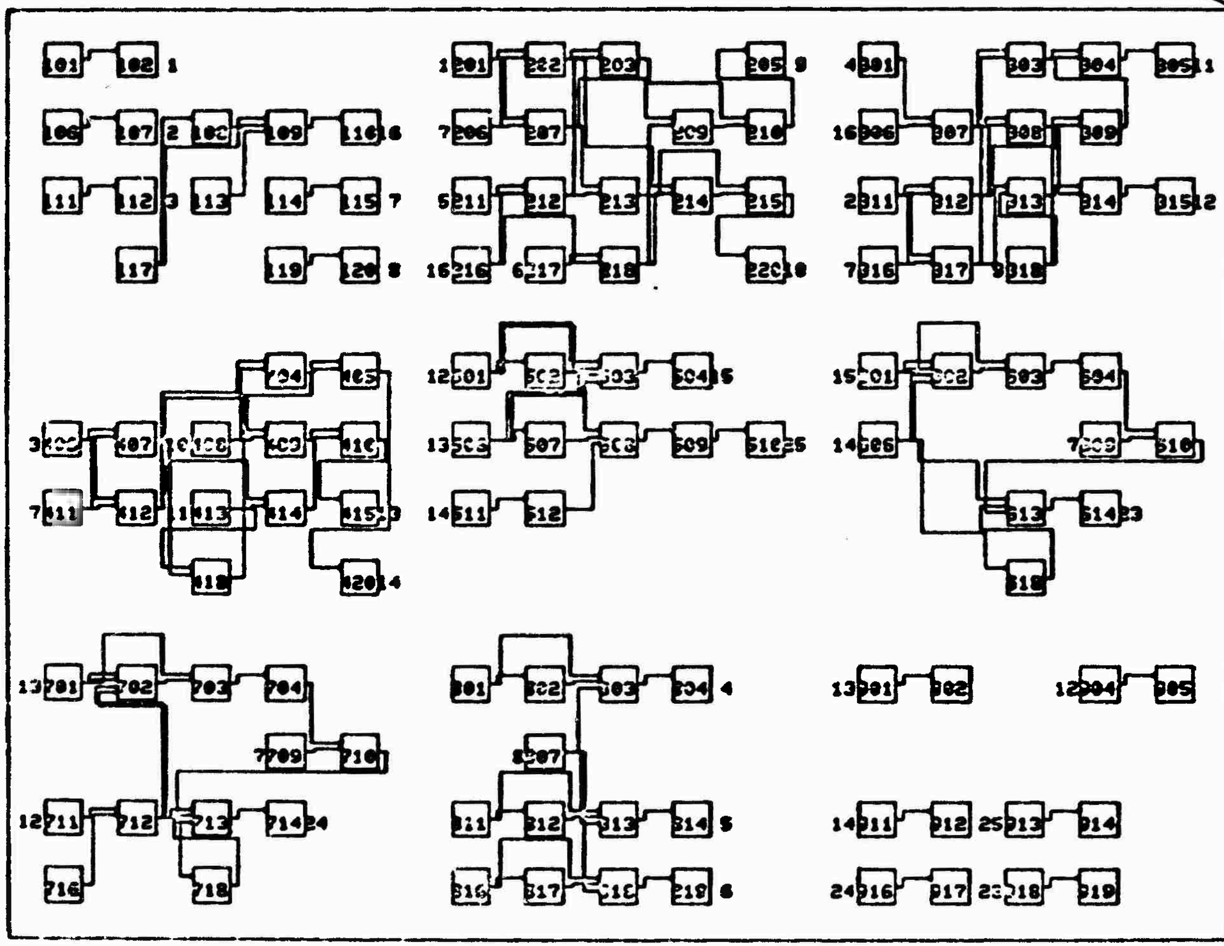
PAGE 8 IPAA
ALGO. 8 1 ACCELERATION SEC.



PAGE 9 IPAA
ALGO. # 1 ACCELERATION SEC.



IPAA ZOOM
ALGO. 8 1 ACCELERATION SEC.



FUNCTIONS AVAILABLE:

INPUT	OUTPUT
TIME	DELAY
CON	GAIN
INTG1	INTG2
+	-
X	/
SINE	COS
ATAN	EXP
ABS	MOD
8HOLD	FGEN1
DISP	CAL
SIMP	ASIN
ACOS	TAN
SQRT	POWER
LOGE	LOG10

KEY OPTIONS:

- •BLOCK DEF
- A•ANNOTATE
- ABCD•PINS
- C•CONNECT
- D•REDRAW
- E•ERASE• ABCDNT
- N•NET
- F•FUNC. DEF
- K•CHECK DIA
- L•LAST DIAG
- M•MOD PAR
- P•PARAM PRT
- R•RECALL
- S•SAVE
- T•TEXT INP
- Z•ZOOM
- 1-9•PAGE
- X•EXECUTE DIAG
- CNTL-C•EXIT

```
OUTPUT ALGO 8 1 ACCELERATION

                  ** IPAA PROCESSOR INFORMATION **
                  1- PROCESSING DESCRIPTION.

                  2- DIAGRAM NAME                     ALGOA.
                  3- START TIME                          1.000
                  4- STOP TIME                          41.000
                  5- TIME INCREMENT                      1.000
                  6- INPUT INTERPOLATION DEGREE      1
                  7- DISPLAY RATIO                    1
                  8- TIME OPTION                     TIME INPUT PROVIDED
                  -------
                  INPUTS.
                  9- INPUT FORMAT                    FORMATTED DATA

                  10- INPUT FILE NO. 1               FILE NAME                    START TIME
                                                     GIMBAL2.                       0.000
                  11- NUMBER OF FIELDS                7
                  12- INPUT                           T 1 2 3 4 5 6
                      REFERENCE                       1 2 3 4 5 6 7
                  13- FIELD                           1 2 3 4 5 6 7
                      WIDTH                          15151515151515
                  OUTPUT
                  ------
                  14- OUTPUT FORMAT                  DEFAULT FORMAT(POT)

                  15- OUTPUT FILE NO. 1              FILE NAME
                  16- OUTPUT FILE NO. 2
                  17- OUTPUT FILE NO. 3
                  18- OUTPUT FILE NO. 4
                  20- SAVE PIF IN FILE
                  21- RESTORE PIF FROM FILE
                  22- RETURN TO DIAGRAM
                  30- EXECUTE
```

SECTION B.3

ALGORITHM NO. 2 CONVENTIONAL CODE LISTING

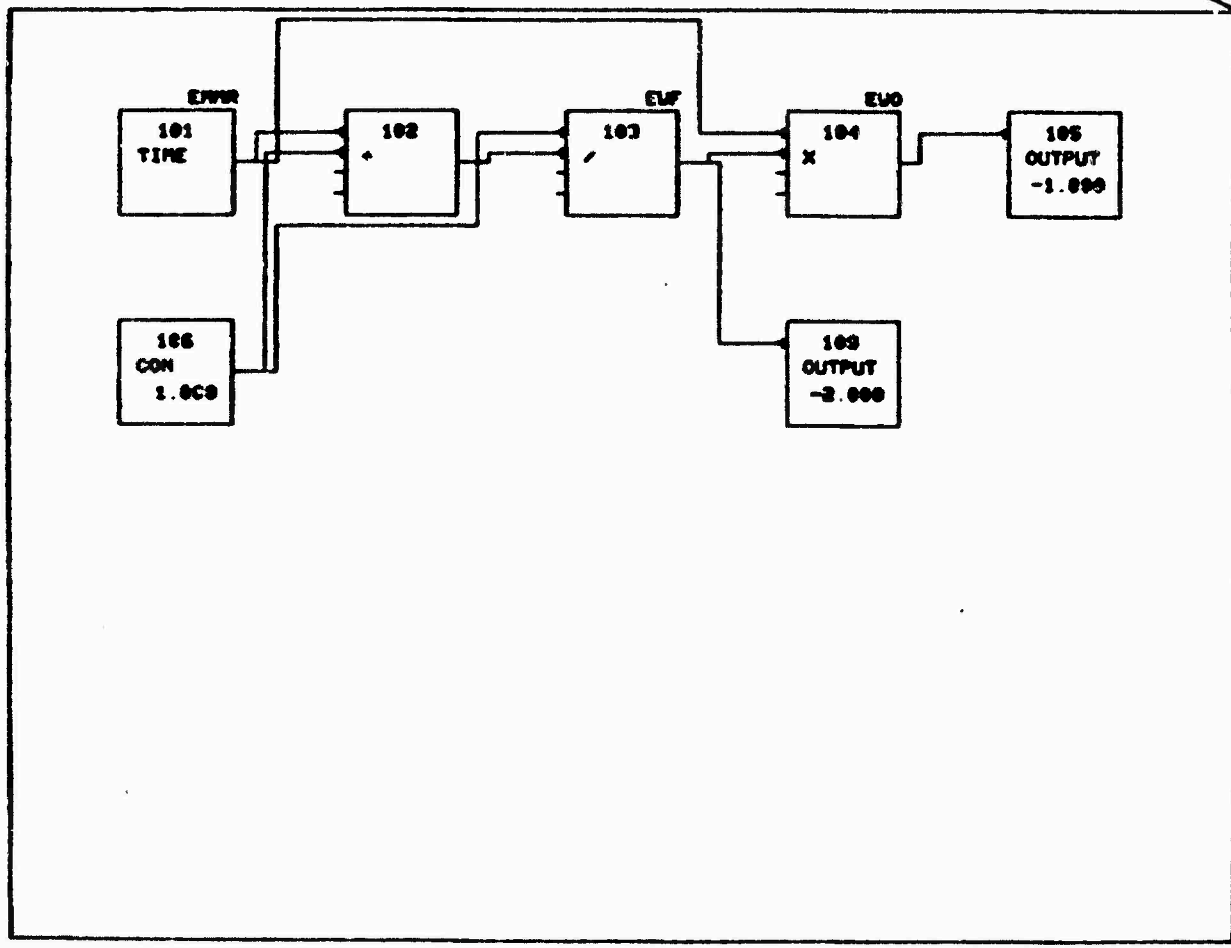
```
JAG-SDP*MASSRAT(1).PROG1
          LUN1=6
          LUN2=9
          LUN3=10
          WRITE (6,10)
   10     FORMAT(' ENTER START, STOP, INCREMENT')
          READ (5,11) EMMRB,EMMRE,EMMRI
   11     FORMAT ()
C
          ICT=(EMMRE-EMMRB)/EMMRI +1
          EMMR=EMMRB
          WRITE (6,100)
  100     FORMAT (7X,'EMMR',11X,'OPBMMR',10X,'PCTPBF',10X,
         -'FPBMMR',10X,'RPBMMR',10X,'PCTPBO')
          DO 1000 I=1,ICT
          EWF=1./(EMMR + 1.)
          EWO = EMMR * EWF
C         C         OX RICH PRE-BURNER
          PBWF = .75 - EWO
          OPBMR=EWO/PBWF
          PCTPBF=PBWF/EWF
C         FUEL RICH PRE-BURNER
          PBWO=.75 - EWF
          FPBMR=PBWO/EWF
          PCTPBO=PBWO/EWO
          RPBMR=1./FPBMR
C
          WRITE (LUN1,300) EMMR,OPBMR
  300     FORMAT (1X,E14.6,',',E14.8)
          WRITE (LUN2,300) EMMR,FPBMR
          WRITE (LUN3,300) EMMR,RPBMR
          WRITE (6,200) EMMR,OPBMR,PCTPBF,FPBMR,RPBMR,PCTPBO
  200     FORMAT (1X,6(2X,E14.8))
 1000     EMMR = EMMR + EMMRI
          END FILE LUN1
```

```
      END FILE LUN2
      END FILE LUN3
      REWIND LUN1
      REWIND LUN2
      REWIND LUN3
      CALL EXIT
      END
```

SECTION B.4

ALGORITHM NO. 2 IPAA DIAGRAMS

PAGE 1 IPAA
ALGO. # 2



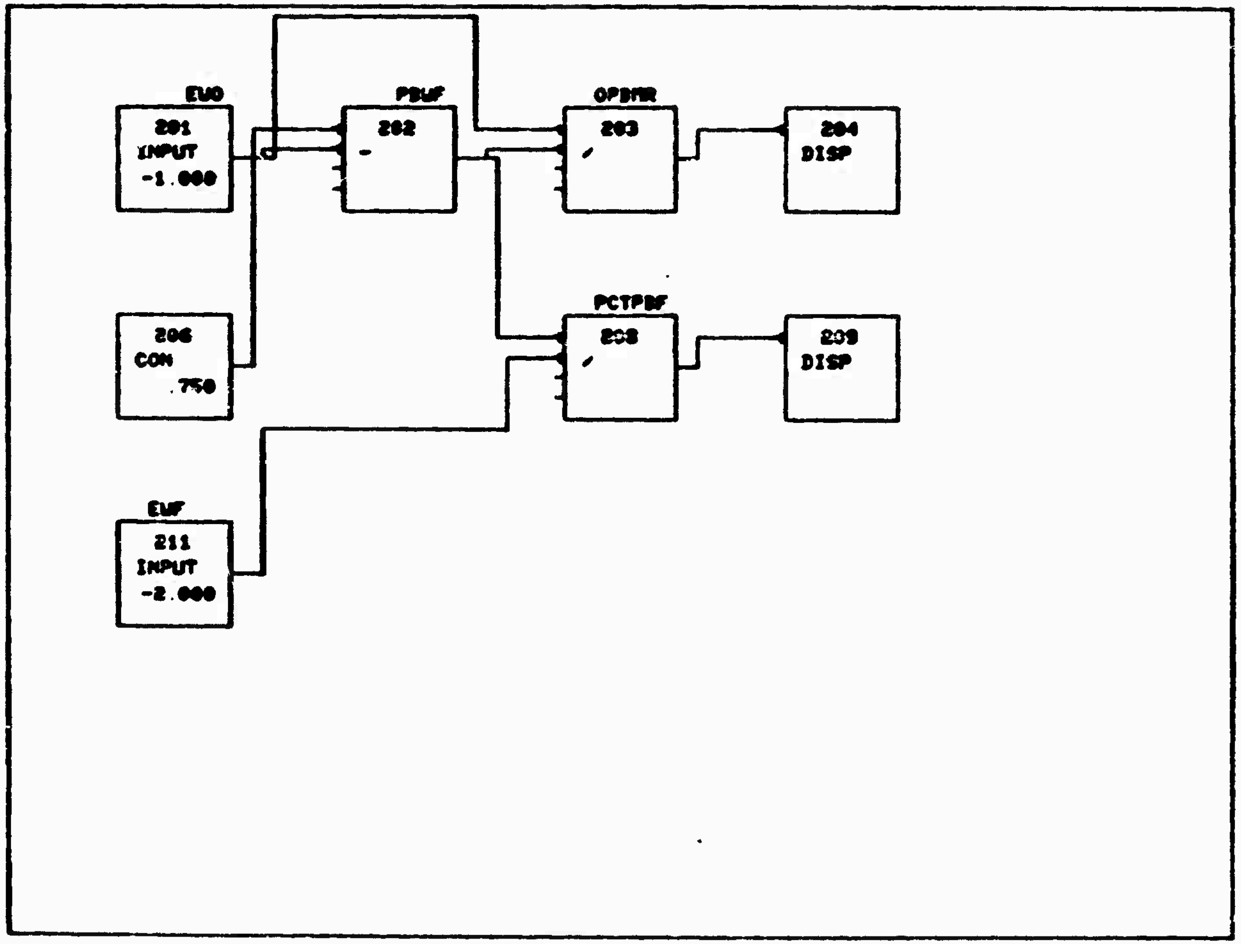
FUNCTIONS AVAILABLE:

INPUT	OUTPUT
TIME	DELAY
CON	GAIN
INTG1	INTG2
+	-
X	/
SINE	COS
ATAN	EXP
ABS	MOD
GHOLD	FGEN1
DISP	CAL
SIMP	ASIN
ACOS	TAN
SQRT	POWER
LOGE	LOG10

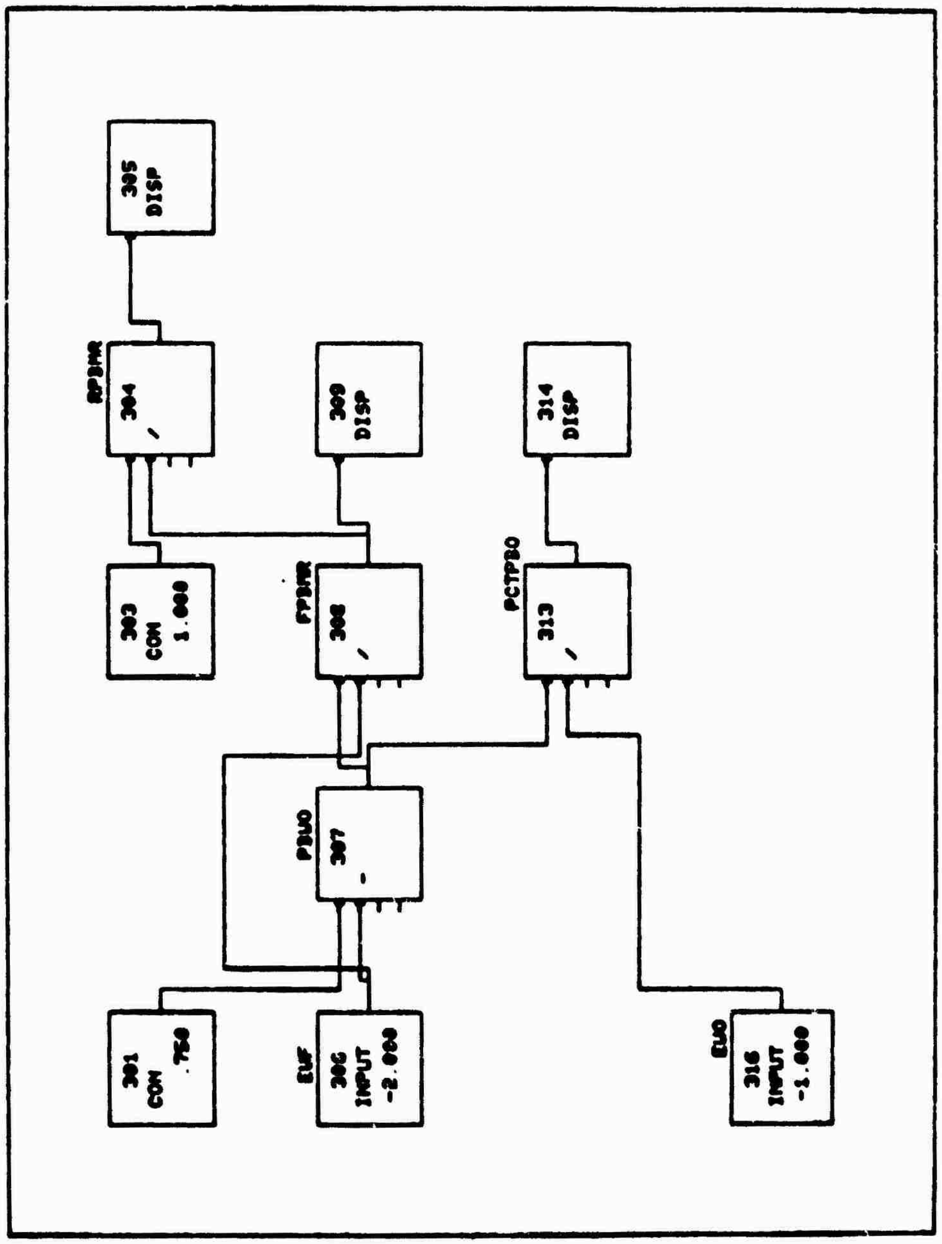
KEY OPTIONS:

- •BLOCK DEF
- A•ANNOTATE
- ABCD•PINS
- C•CONNECT
- D•REDRAW
- E•ERASE• ABCDNT
- N•NET
- F•FUNC. DEF
- K•CHECK DIA
- L•LAST DIAG
- M•MOD PAR
- P•PARAM PRT
- R•RECALL
- S•SAVE
- T•TEXT INP
- Z•ZOOM
- 1-9•PAGE
- X•EXECUTE DIAG
- CNTL-C•EXIT

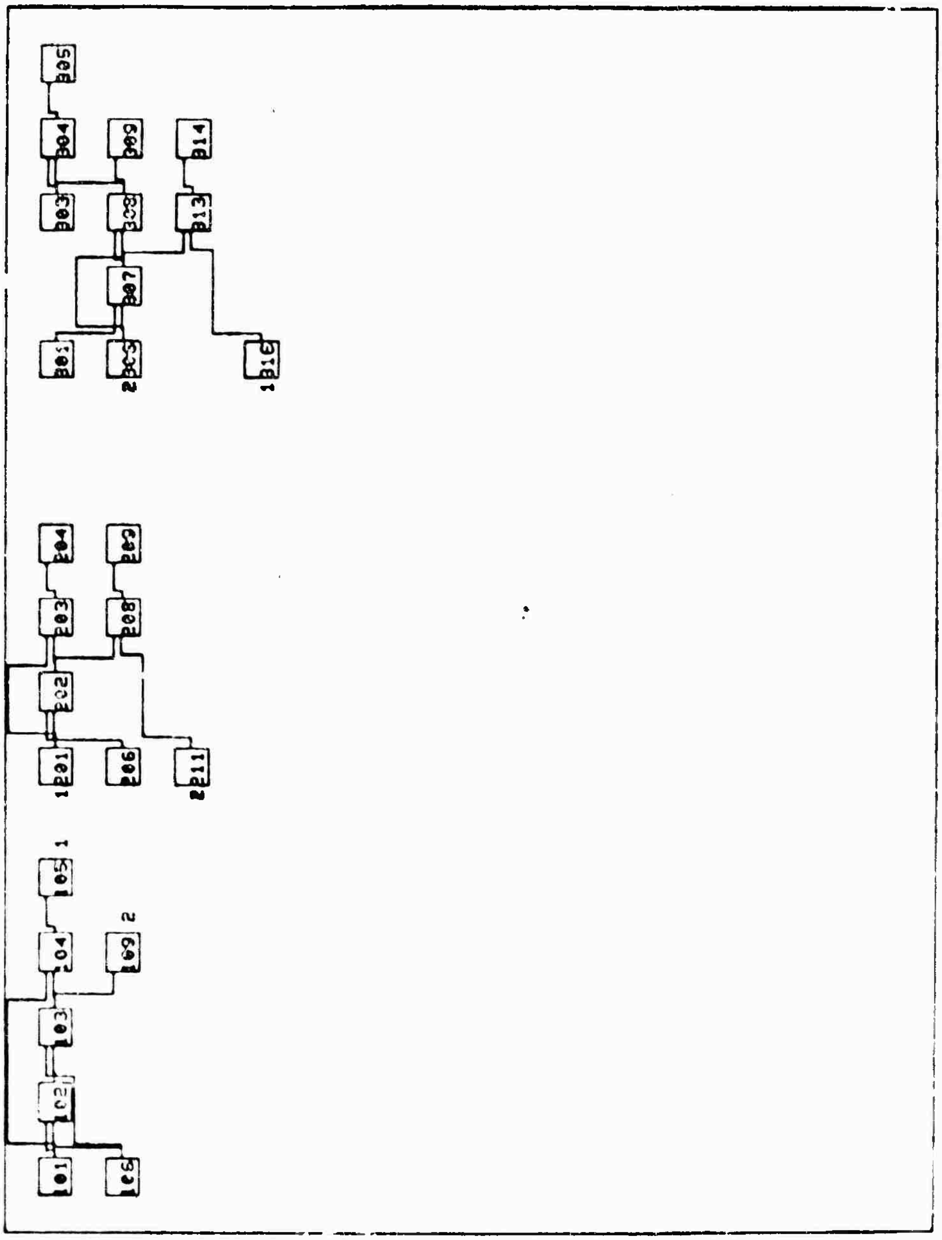
PAGE 8 IPAA
ALGO. 8 2



PAGE 3 IPAA
ALGO. # 2



IPPA ZOOM
ALGO # 2



```
OUTPUT ALGO # 2

                ** IPAA PROCESSOR INFORMATION **
1- PROCESSING DESCRIPTION.

2- DIAGRAM NAME
3- START TIME                        0.000
4- STOP TIME                         3.000
5- TIME INCREMENT                     .100
6- INPUT INTERPOLATION DEGREE     0
7- DISPLAY RATIO                    1
8- TIME OPTION                    TIME INPUT PROVIDED
-------
INPUTS
9- INPUT FORMAT                   DEFAULT FORMAT(POT)
                                  FILE NAME                    START TIME
10- INPUT FILE NO. 1                                             0.000
11- INPUT FILE NO. 2                                             0.000
12- INPUT FILE NO. 3                                             0.000
13- INPUT FILE NO  4                                             0.000
OUTPUT
------
14- OUTPUT FORMAT                 DEFAULT FORMAT(POT)
                                  FILE NAME
15- OUTPUT FILE NO  1
16- OUTPUT FILE NO  2
17- OUTPUT FILE NO. 3
18- OUTPUT FILE NO  4
20- SAVE PIF IN FILE
21- RESTORE PIF FROM FILE
22- RETURN TO DIAGRAM
30- EXECUTE
```

MISSION
of
Rome Air Development Center

RADC plans and conducts research, exploratory and advanced development programs in command, control, and communications (C^3) activities, and in the C^3 areas of information sciences and intelligence. The principal technical mission areas are communications, electromagnetic guidance and control, surveillance of ground and aerospace objects, intelligence data collection and handling, information system technology, ionospheric propagation, solid state sciences, microwave physics and electronic reliability, maintainability and compatibility.

AMERICAN REVOLUTION BICENTENNIAL 1776 · 1976